

7-8 1981

informatyka



WYDAWNICTWO NOT SIGMA

Do Autorów

Od dłuższego już czasu nie zamieszczamy na łamach pojawiających się dawniej regularnie — odredakcyjnych zaleceń dla potencjalnych autorów. Doszliśmy bowiem do wniosku, że precyzyjne wskazówki dotyczące objętości, formy czy konstrukcji raczej zniechęcają do pisania. Brak takich informacji okazał się jednak również wyjściem nie najlepszym, autorzy bowiem często pytają o podstawowe warunki przyjęcia materiału do druku.

Spróbujemy zatem w mniej zawiły sposób przedstawić zasady, jakimi kierujemy się przy kwalifikacji artykułu oraz warunki formalne, jakie powinien spełniać maszynopis.

Podstawowym kryterium merytorycznym materiału (nie licząc głównego kryterium — oceny z punktu widzenia dotychczasowej wiedzy informatycznej) jest społeczna użyteczność zawartych w tekście rozważań i informacji. To w końcu oczywiste — jesteśmy powołani przede wszystkim do tego, aby służyć czytelnikom. W pierwszej kolejności liczyć się musi zatem interes społeczny. O tym winni zresztą pamiętać przede wszystkim autorzy.

Objętość tekstu powinna być wyznaczona przez społeczną wagę tematu, jego konstrukcja — przez wymóg czytelnego przekazu, zaś forma — przez dostosowanie języka i sposobu argumentacji do możliwości percepcyjnych przeciętnego odbiorcy. To truizmy, niemniej — jak praktyka wskazuje — warte przypomnienia. Autorzy najczęściej nie zadają sobie trudu, by wprowadzić czytelnika w zawiłości swoich specjalistycznych wywodów, nie biorą pod uwagę miejsca publikacji artykułu, tzn. charakteru naszego czasopisma.

Jeśli zaś chodzi o wymogi formalne — nie są duże. Co do objętości — maszynopis nie powinien przekraczać 12 znormalizowanych stron (30 wierszy z 60 znakami w wierszu). Jeśli jednak interesujący, zgodny z profilem pisma temat wymaga większej objętości, to oczywiście względy formalne nie stoją na przeszkodzie. Rysunki techniczne, dołączane do tekstu, są w Redakcji przygotowywane przez wykwalifikowanego kreślarza, nie muszą być zatem dostarczane na kalce, ważne natomiast, aby były całkowicie czytelne, przejrzyste i zachowujące konieczne proporcje.

Pożądany jest również dodatkowy materiał ilustracyjny (kontrastowe zdjęcia czarno-białe, wydruki, itp.), jeśli wzbogaca on bądź lepiej precyzuje wywód autora.

Maszynopis powinien być nadsyłany w dwóch egzemplarzach (oryginał + kopia), z załączeniem aktualnych adresów, a także numerów telefonów, umożliwiających szybki kontakt. Autorów artykułów problemowych, którzy nie zamieszczali swych prac w ciągu dwóch ostatnich lat, prosimy ponadto o załączenie zdjęcia oraz życiorysu.

Redakcja

WYDAWNICTWO
NACZELNA ORGANIZACJA TECHNICZNA
CZASOPISM I KSIĄŻEK TECHNICZNYCH
SIGMA
ul. Świętokrzyska 14a
00-950 Warszawa
skrytka pocztowa 1004

Redakcja: 00-041 Warszawa, ul. Jasna 14/16, pokój 326, tel. 27-71-40, dyżury redakcji 10.00—13.00

Zakł. Graf. „Tamka". Zam. 172. Obj. 5 ark. druk. Nakład 6300 egz. L-103.

Cena egzemplarza zł 30.— INDEKS 36124 Prenumerata roczna zł 360.—

Carlson P. K.: Zastosowanie języka CDL2 do konstrukcji oprogramowania dla MERY 400

INFORMATYKA 1981, nr 7-8, s. 4

Charakterystyka języka CDL2 oraz sposobu jego wykorzystywania do tworzenia oprogramowania minikomputera MERA 400 przy użyciu komputerów R-32 lub IBM 360/370. Omówiono ogólną filozofię języka ilustrowaną przykładową strukturą programu oraz podano podstawowe informacje o możliwościach praktycznego wykorzystania proponowanych rozwiązań.

Карльсон П. К.: Применение языка CDL-2 для конструкции программного обеспечения для MERY 400

ИНФОРМАТИКА 1981, № 7—8, стр. 4

Характеристика языка CDL-2 и способ его использования для создавания программного обеспечения малой вычислительной машины MERA 400 при использованию вычислительных машин Р-32 или IBM 360/370. Обсуждается общая философия языка иллюстрированная примерной структурой программы и даются информации о возможностях практического использования предлагаемых решений.

Bieleninik E.: Nowe zastosowania teleinformatyki

INFORMATYKA 1981, nr 7-8, s. 8

Przegląd nowych zastosowań teleinformatyki na przykładzie systemów zrealizowanych we Francji w oparciu o uniwersalne sieci transmisji danych. Podano charakterystykę wybranych rodzajów zastosowań oraz użytego do ich realizacji sprzętu.

Беленинник Э.: Новое применение дистанционной вычислительной техники

ИНФОРМАТИКА 1981, № 7—8, стр. 8

Просмотр новых применений дистанционной вычислительной техники на примере систем реализованных во Франции на основе универсальной сети передачи данных. Дается характеристика избранных видов применений и используемого для реализации оборудования.

Maciaszek L. A.: Słownik-skorowidz w procesie powstawania i utrzymywania baz danych

INFORMATYKA 1981, nr 7-8, s. 11

Uzasadnienie potrzeby tworzenia oraz charakterystyka funkcji pakietu programowego słownika-skorowidza danych w procesie powstawania (projektowania, programowania i zakładania) oraz utrzymywania (eksploatacji i rozwoju) baz danych. Podano przykłady istniejących rozwiązań zagranicznych oraz wskazano na potrzebę podjęcia tego rodzaju prac w Polsce.

Мациашек Л. А.: Словарь-указатель в процессе образования и сохранения баз данных

ИНФОРМАТИКА 1981, № 7—8, стр. 11

Обоснование потребности создания и характеристика функций программного пакета словаря-указателя данных в процессе образования (проектирования, программирования и закладывания), а также сохранения (эксплуатации и развития) баз данных. Даются примеры существующих зарубежных решений и указывается на целесообразность начала такого рода работ в Польше.

Matwiejczuk J.: ODRA 1300 jako system pośredniczący w programowaniu mikroprocesorów

INFORMATYKA 1981, nr 7-8, s. 13

Charakterystyka konstrukcji oraz warunków eksploatacji programu przeznaczonego do tworzenia oprogramowania mikroprocesorów typu INTEL 8080 przy użyciu komputera serii ODRA 1300. Bardziej szczegółowo omówiono podstawowe moduły tego programu: makrogenerator, asembler i emulator.

Матвейчук Й.: ODRA 1300 как система посредничающая в программе микропроцессоров

ИНФОРМАТИКА 1981, № 7—8, стр. 13

Характеристика конструкции и условий эксплуатации программы предназначенной для создавания программного обеспечения микропроцессоров типа INTEL 8080 в случае использования вычислительной машины серии ODRA 1300. Более подробно обсуждаются основные модули этой программы: макрогенератор, асемблер и эмулятор.

Jaegermann T.: Ochrona danych obowiązuje

INFORMATYKA 1981, nr 7-8, s. 16

Omówienie głównych problemów ochrony danych w systemach informatycznych. Opierając się na bogatej literaturze zagranicznej autor wskazuje na wzrastającą wagę ochrony danych, podając przykładowe sposoby stosowanych zabezpieczeń i przezwyciężania występujących trudności, a także wskazuje na konieczność bardziej intensywnego zajęcia się tym zagadnieniem w Polsce.

Ягерманн Т.: Обязывает охрана данных

ИНФОРМАТИКА 1981, № 7—8, стр. 16

Обсуждение главных проблем охраны данных в вычислительных системах. Опираясь на богатой зарубежной литературе автор указывает на растущее значение охраны данных. Даются примерные способы используемых обеспечений и способы преодоления возникающих трудностей. Подчеркивается также необходимость более интенсивной разработки этого вопроса в Польше.

Carlson P. K.: Applying the CDL2 language for the construction of MERA 400 software

INFORMATYKA 1981, No 7-8, p. 4

Characteristics of the CDL2 language and its application for MERA 400 minicomputer software creation using R-32 or IBM 360/370 computers. Discussed general philosophy of the language, illustrated with exemplary structure of the program, and presented basic information about possibilities of proposed solutions practical utilization.

Carlson P. K.: Anwendung der CDL2 Sprache für die Softwarekonstruktion von MERA 400

INFORMATYKA 1981, Nr. 7-8, S. 4

Eine Charakteristik der CDL2 Sprache und ihrer Ausnutzung für die Softwareherstellung von MERA 400 Kleinrechner mit Hilfe der R-32 und IBM 360/370 Rechner. Es wurden die allgemeine Philosophie dieser Sprache, abgebildet mit Musterstruktur des Programms, und die wichtigsten Informationen über Möglichkeiten der praktischen Ausnutzung der vorgeschlagenen Lösungen besprochen.

Bieleninik E.: New applications of teleinformatics

INFORMATYKA 1981, No 7-8, p. 8

An overview of new teleinformatics applications on the example of data processing systems realized using universal data processing network in France. Presented characteristics of selected application types and of hardware used for their realization.

Bieleninik E.: Neue Anwendungen der Teleinformatik

INFORAMATYKA 1981, Nr. 7-8, S. 8

Eine Übersicht der neuen Teleinformatikanwendungen mit Beispielen der in Frankreich realisierten Systeme, die bestehende universelle Datenübertragungsnetze ausnützen. Es wurde eine Charakteristik der ausgewählten Typen von Anwendungen und der zu ihrer Realisation eingesetzten technischen Mittel angegeben.

Maciaszek L. A.: Dictionary-directory in data bases creation and maintenance process

INFORMATYKA 1981, No 7-8, p. 11

Justification of creating necessity as well functional characteristics of program package for data dictionary-directory in data bases creation (designing, programming, initiating) and maintenance (operating and developing) process. Presented examples of existing foreign solutions and pointed out the necessity of undertaking similar work in Poland.

Maciaszek L. A.: Wörterbuch-Verzeichnis im Entstehungs- und Erhaltungsprozess der Datenbasen

INFORMATYKA 1981, Nr. 7-8, S. 11

Eine Bedürfnisbegründung der Schaffung und eine Funktionscharakteristik eines Programmpakets für das im Entstehungs- (Projektierung, Programmierung und Errichtung) und Erhaltungsprozess (Betrieb und Entwicklung) der Datenbasen benötigte Wörterbuch-Verzeichnis. Es wurden Beispiele der heutigen ausländischen Lösungen angegeben, sowie Bedürfnis für die Aufnahme von derartigen Arbeiten in Polen betont.

Matwiejczuk J.: ODRA 1300 as intermediary system for microprocessor programming

INFORMATYKA 1981, No 7-8, p. 13

Characteristics of construction and operating conditions of the program, which is devoted for INTEL 8080 class microprocessor software creation using ODRA 1300 computer series. Discussed with more detail basic modules of the program: macrogenerator, assembler and emulator.

Matwiejczuk J.: ODRA 1300 als Vermittlungssystem für Mikroprozessorenprogrammierung

INFORMATYKA 1981, Nr. 7-8, S. 13

Eine Charakteristik der Konstruktion und der Betriebsbedingungen eines Programms für die Softwareherstellung der INTEL 8080-ähnlichen Mikroprozessoren mit Hilfe der ODRA 1300 Rechnerserie. Mit grösseren Einzelheiten wurden die grundlegenden Module dieses Programms: Makrogenerator, Assembler und Emulator besprochen.

Jaegermann T.: Data protection is obligatory

INFORMATYKA 1981, No 7-8, p. 16

Presentation of main problems of data protection in edp-systems. Based on rich foreign references pointed out to growing significance of data protection problem giving examples of applied protections and some substantial difficulties, as well to necessity of more intensive dealing with this problem in Poland.

Jaegermann T.: Datenschutz ist Pflicht

INFORMATYKA 1981, Nr. 7-8, S. 16

Eine Besprechung der wichtigsten Probleme des Datenschutzes in EDV-Systemen. Auf Grund der umfangreichen ausländischen Literatur zeigt der Autor die wachsende Bedeutung des Datenschutzes an, gibt Beispiele von verwendeten Sicherungen und Überwindung der Schwierigkeiten an, sowie weist eine Notwendigkeit der intensiveren Erforschung dieses Problems in Polen an.

Informatyka

zastosowania w gospodarce, technice i nauce

Nr 7-8
MIESIĘCZNIK
1981
ROK XVI
Lipiec — sierpień

ORGAN KOMITETU INFORMATYKI, MINISTERSTWA NAUKI, SZKOLNICTWA WYŻSZEGO I TECHNIKI ORAZ KOMITETU NAUKOWO-TECHNICZNEGO NOT DS. INFORMATYKI

W NUMERZE: Strona

PIOTR K. CARLSON
Ośrodek Obliczeniowy Instytutu Informatyki
Uniwersytet Warszawski

Zastosowanie języka CDL2 do konstrukcji oprogramowania dla MERY 400

Jakkolwiek cechy użytkowe systemu OS/360 są stałym tematem gorzkich żartów krążących wśród programistów, to jednak stanowi on środowisko bardziej sprzyjające pracy nad dużymi programami niż SOM-3, fabryczny system operacyjny minikomputera MERA 400. Jest to spowodowane ubóstwem sprzętowym typowej konfiguracji MERY 400 oraz niewyrafinowaniem jej systemu plików, odczuwalnym zwłaszcza przez tych użytkowników SOM-3, którzy nie korzystają z edytora EDM. W tym stanie rzeczy programiści nie uważający BASIC ani FORTRAN za języki implementacyjne mogą być zainteresowani schematem translacji skrośnej pomiędzy komputerem R-32 a MERĄ 400.

Artykuł ten stanowi opis takiego schematu, w którym językiem programowania jest CDL2 pochodzący od Kostera [5]. Język ten jest prawie całkowicie w Polsce nieznany. Oprócz ciekawych cech samego języka (m.in. wspomaganie konstrukcji przenaszalnego a zarazem zorientowanego maszynowo oprogramowania), zainteresowanie może budzić istnienie jego kompilatorów generujących kod dla komputerów IBM 360/370, MERY 400 oraz PDP-11, a więc architektur stosunkowo w Polsce popularnych. W Instytucie Informatyki UW autor stosuje CDL2 w dydaktyce oraz do konstrukcji i rozwoju dalszych implementacji, m.in. dla MERY 400.

Język programowania CDL2 jest następcą metatranslatora o nazwie CDL (Compiler Description Language) [6]. Autorami pierwszego kompilatora byli H. M. Stahl i H. Feuerhahn z Technicznego Uniwersytetu w Berlinie Zachodnim. Podstawowym zastosowaniem CDL2 jest konstrukcja kompilatorów. Szczególne cechy tego języka [8] czynią go wszakże przydatnym do szeroko pojętego programowania systemowego w dziedzinach, takich jak systemy operacyjne [1] czy oprogramowanie sieci komputerowych [3]. Wiele zastosowań opartych jest o schemat translacji skrośnej do produkcji oprogramowania dla małych maszyn, a także mikrokomputerów [4]. Nie dają się też pominąć walory dydaktyczne języka, który wspomaga zasady dobrej praktyki i programowania strukturalnego. W krajach RWPG CDL2 jest stosowany jedynie na Węgrzech [2], choć CDL1 został zaimplementowany w NRD na maszynie ROBOTRON 4000 [7].

OGÓLNA FILOZOFIA JĘZYKA CDL2

Rozwiązania przyjęte w CDL2 ukierunkowane są na dwie grupy problemów. Pierwsza z nich to sprawy związane z przenaszalnością oprogramowania systemowego, z konieczności ściśle związanego z konkretną maszyną czy systemem operacyjnym. Druga sprawa to problemy związane z konstrukcją i konserwacją dużych i złożonych programów (np. kompilatorów), a wynikające właśnie z ich wielkości i złożoności. Są to przede wszystkim kwestie czytelności i modyfikowalności.

CDL2 jest językiem o tzw. otwartym jądrze, to znaczy rozszerzalnym semantycznie przez zastosowanie mechanizmu pożyczania z innego języka, najczęściej asemblera. Pożyczanie odbywa się za pomocą makrodefinicji. Programista używa zatem dwu języków: języka źródłowego, czyli CDL2, oraz języka docelowego, którym jest np. makroasembler.

CDL2 zawiera jedynie minimum pojęć pierwotnych. Od strony algorytmicznej — struktury sterowania buduje się w sposób zbliżony do LISP wykorzystując pojęcia sukcesu i porażki algorytmu. Od strony danych — jedynymi typami znanymi w CDL2 są: nieinterpretowane słowo maszyny i wektor słów (tablica jednowymiarowa), a zatem praktycznie tylko środki służące rezerwacji pamięci. Wszystkie operacje łącznie z arytmetyką i wejściem/wyjściem programista definiuje sam na poziomie docelowym dostarczając odpowiednich makrodefinicji, takich jak „dodaj", „pomnóż przez 8", „pisz znak", „czytaj znak". Poziom źródłowy CDL2 zawiera tylko narzędzia do strukturalizacji programu — konstrukcji i abstrakcji, a jądro semantyczne pozostaje puste — do określenia na poziomie docelowym. Oznacza to, że w programie można przeprowadzić wyraźny podział na część zależną od maszyny (w języku docelowym) i część sformułowaną wyłącznie w CDL2, która jest gwarantowana jako maszynowo niezależna. Gwarancja ta nie jest ryzykowna z uwagi na ogromną prostotę pojęć występujących w „czystym" CDL2.

Technika przenoszenia programów pomiędzy maszynami dla których istnieją odpowiednie kompilatory CDL2 polega na wymianie ciał makrodefinicji. Powoduje to, że koszt przeniesienia jest zawsze niezerowy. Korzyścią, która zwykle wyrównuje (w stosunku do innych języków) poniesione nakłady jest uniknięcie częstych nieporozumień związanych z identyfikacją części programu pozornie i faktycznie zależnych od maszyny. Zewnętrzna specyfikacja algorytmu jest identyczna dla makro i dla reguły CDL2, zatem zależnie od mocy języka docelowego niektóre makrodefinicje mogą być zastępowane regułami i odwrotnie [1]. Umożliwia to najdalej idącą adaptację programu. Każda instrukcja maszyny staje się dostępna w optymalizacji kluczowych fragmentów programu. Za cenę braku standardowych typów danych uzyskuje się zatem maksymalną swobodę w ich implementacji. Oczywiście podejście takie jest opłacalne jedynie w przypadku stosunkowo dużych programów, gdzie koszt implementacji np. typu „integer" jest mały w porównaniu z kosztem konstrukcji całego programu. CDL2 zdecydowanie nie jest językiem do szybkiego pisania krótkich, podręcznych programów, ale dodatkowy wysiłek włożony w definiowanie pojęć zwykle dostępnych jako standardowe procentują w dużych programach lepszymi możliwościami manipulacji.

Manipulacja programem należy do drugiej grupy problemów — związanych z „dużym programowaniem". CDL2 dysponuje tu co najmniej dwoma poważnymi atutami: sta-



Mgr Piotr K. CARLSON ukończył w 1975 r. Wydział Matematyki i Mechaniki Uniwersytetu Warszawskiego. Od 1976 r. pracuje w Instytucie Informatyki UW, początkowo na stanowisku asystenta, a obecnie specjalisty w Ośrodku Obliczeniowym Instytutu. W roku akademickim 1978—1979 odbywał staż na Uniwersytecie w Nijmegen (Holandia). Zajmuje się językami programowania, narzędziami wspomagającymi oprogramowanie, grami komputerowymi.

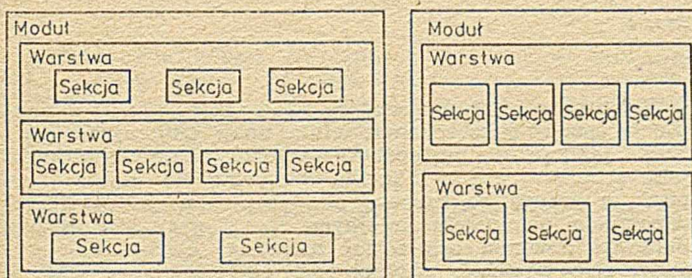
[1]) Algorytm jest tu odpowiednikiem pojęcia procedury w innych językach. Makrodefinicje to algorytmy wyrażone w języku, z którego pożyczamy; natomiast reguły to algorytmy wyrażone w CDL2.

tyczna rozstrzygalność wielu własności programu pozwala kompilatorowi dokonać globalnych optymalizacji oraz sprawdzenia poprawności takich aspektów programu jak przepływy informacji i efekty uboczne algorytmów. Ma to kapitalne znaczenie przy modyfikowaniu programu, zarówno dlatego, że kompilator jest w stanie wykryć wiele błędów w logice programu, takich jak użycie nieokreślonych wartości, czy też nieświadome zaniedbanie wyniku częściowego, ale też dlatego, że kompilator wymusza niejako na programiście przestrzeganie dobrego stylu programowania. Zastosowanie globalnej analizy programu daje w efekcie luźniejszy związek między strukturą tekstu źródłowego a strukturą (i efektywnością!) kodu wynikowego. Kompilator może podjąć decyzję o sposobie implementacji każdej procedury stosownie np. do jej użycia (często używane — jako podprogramy zamknięte, rzadko używane — jako otwarte, w ogóle nie używane — usunięte z programu). Może również dokonać transformacji programu zbliżonych do tych, które robi programista poświęcający dobry styl dla większej efektywności kodu, a więc: zastępowanie rekursji iteracją, dzielenie zmiennej roboczej pomiędzy dwa koncepcyjnie różne obiekty nie współistniejące ze sobą w czasie, stosowanie skoków dla uniknięcia powtórzenia fragmentu tekstu programu. W CDL2 kompilator ma dostatecznie dużo informacji o programie, aby uwolnić programistę od konieczności stosowania tych chwytów, ponieważ może zastosować je sam. Dla programisty nadrzędnym celem jest uzyskanie programu czytelnego, o prawidłowej strukturze, optymalnego w sensie doboru i implementacji struktur danych oraz strategii globalnych. Lokalna optymalizacja nie powinna obciążać programisty w czasie konstrukcji programu i powinna być dokonywana ręcznie jedynie w fazie adaptacji uruchomionego już programu.

STRUKTURA PROGRAMU W CDL2

CDL2 jest językiem modularnym. Program złożony jest ze zbioru definicji algorytmów (makro i reguł), zmiennych, stałych, statycznie alokowanych jednowymiarowych tablic oraz trzech sekwencji wywołań algorytmów. Trzy sekwencje to: wywołania inicjalizacyjne (PRELUDE), wywołania stanowiące główną akcję programu (ROOT) i zakończenie (POSTLUDE). Wszystkie obiekty wchodzące w skład programu są identyfikowane za pomocą nazw. W CDL2, jako języku o pustym jądrze, nie ma żadnych nazw standardowych; każdy obiekt musi być skonstruowany i nazwany przez programistę. W początkowym okresie jest to kłopotliwe, ale po nabraniu wprawy nazwy z korzyścią przejmują funkcję komentarzy.

Strukturę programu należy omawiać na dwu poziomach: poziomie modułów i poziomie algorytmów. Struktura algorytmu (pojedynczej reguły) jest wymuszona składnią. Strukturę programu wyznaczają relacje widzialności nazw obiektów. Dzielą one program w sposób podany na rys. 1.



Rys. 1

Moduł jest jednostką kompilacji. Program może być złożony z jednego lub więcej niezależnie kompilowanych modułów. Moduł jest zbudowany z jednej lub więcej (typowo — trzech) warstw zawierających sekcje. Wszystkie definicje obiektów występują wyłącznie wewnątrz sekcji i w nich obiekty te są lokalne. Algorytmy i stałe mogą zostać udostępnione innym sekcjom znajdującym się w tej samej warstwie, lub w warstwie o 1 wyższej, mogą też być widoczne z zewnątrz modułu. Udostępnienie obiektu każdemu z tych trzech otoczeń sekcji następuje zawsze explicite, przez wymienienie jego nazwy na odpowiedniej liście. Również explicite muszą być wymienione w nagłówku sekcji wszystkie obiekty używane wewnątrz niej a pochodzące z innej sekcji.

Ideologia związana z taką właśnie strukturalizacją programu jest następująca: kolejne warstwy są realizacją abstrakcyjnych maszyn o prymitywach na coraz wyższym poziomie. Te prymitywne algorytmy są udostępniane wyższej warstwie, gdzie traktowane są tak, jak zwykło się traktować tzw. procedury standardowe. Sekcje grupują definicje np. składające się na realizację struktury danych, obsługę wejścia/wyjścia czy zbiór operacji charakteryzujących pewien typ — w warstwie dolnej, a w górnej — ogólne schematy działania poszczególnych części programu, np. w przypadku kompilatora — analiza syntaktyczna, semantyczna, generacja kodu.

Ograniczenie widzialności pasywnych elementów struktur danych (zmiennych i tablic[2])) do ich macierzystych sekcji ma na celu lokalizację efektów ubocznych.

Algorytmy mogą mieć parametry, zwane afiksami. Wartości przekazywane za pomocą parametrów odpowiadają zawsze jednemu słowu maszyny i nigdy nie podlegają interpretacji w sferze CDL2. W CDL2 opisuje się jedynie strukturę wywołań i przepływu tych wartości pomiędzy wywoływanymi algorytmami. Interpretacji można dokonać oczywiście jedynie wewnątrz makrodefinicji, bowiem tam dysponujemy środkami pożyczonymi w tym celu ze środowiska, języka wewnętrznego konkretnej maszyny.

W programie występują zatem dwa zasadniczo różne rodzaje algorytmów: reguły, których ciała zapisane są w CDL2 i makrodefinicje zbudowane ze stałych tekstów oraz nazw (afiksów, zmiennych, stałych, tablic) zastępowanych przy generacji kodu odpowiednio do modelu wykonawczego przyjętego na danej maszynie.

```
      $ASS LO CSL
      $EXE CDL2CO,BI
OPTIONS SET FOR THIS COMPILATION:
      LIST, NODICT

--- OUTPUT OF CDL2-COMPILER / INFORMATIKA  KU NIJMEGEN / VERSION 8.1 (81.19.02) ---

 'MODULE' PODSTAWOWE OPERACJE NA MERZE 400.

 'LAYER' L.

 'SECTION' PODSTAWOWA ARYTMETYKA NA MERZE 400.

 'EXPORT' USTAL, ZWIEKSZ O 1, ODEJMIJ, POMNOZ, PODZIEL, ZANIEDBAJ,
          JESLI ROWNE, JESLI MNIEJSZE.

 'FUNCTION' USTAL +X> +>Y =
          "$L   LWI,1," Y
          "$L   RWD,1," X .

 'FUNCTION' ZWIEKSZ O 1 +>X> =
          "$L   IBD," X .

 'FUNCTION' ODEJMIJ +>X +>Y +Z> =
          "$L   LWI,1," X
          "$L   SWI,1," Y
          "$L   RWD,1," Z .

 'FUNCTION' POMNOZ +>X +>Y +ILOCZYN MODULO 2 DO 15> =
          "$L   LWI,2," X
          "$L   MWD,"   Y
          "$L   RWD,2," ILOCZYN MODULO 2 DO 15 .

 'FUNCTION' PODZIEL +>X +>Y +ILORAZ> +RESZTA> =
          # CIALO "PODZIEL" CZYTELNIK ZECHCE NAPISAC JAKO CWICZENIE #

 'FUNCTION' ZANIEDBAJ +>X =
          "$L* ZANIEDBAJ " X .

 'TEST' JESLI ROWNE +>X +>Y =
          "$L   LWI,1," X
          "$L   CWI,1," Y
          "$L   JND,"

 'TEST' JESLI MNIEJSZE +>X +>Y =
          "$L   LWI,1," X
          "$L   CWI,1," Y
          "$L   BBD,0,#0800"
          "$L   UJD,"

 'ENDSEC' PODSTAWOWA ARYTMETYKA NA MERZE 400.

 !EJECT!
```

Rys. 2

[2]) Stałe, należące również do tej kategorii można jednak traktować jako szczególny przypadek funkcji i dlatego dopuszcza się ich udostępnianie.

Rys. 2 i 3 dają przykład programu złożonego z dwóch modułów. Pierwszy z nich definiuje podstawowe operacje dla typów „małe nieujemne liczby całkowite” i „znaki”. Operacje te są wyrażone za pomocą makrodefinicji i udostępnione potencjalnemu środowisku modułu.

```
'SECTION' PODSTAWOWE WEJSCIE WYJSCIE NA MERZE 400.

'EXPORT' WCZYTAJ ZNAK, WYPISZ ZNAK, SP, EOT.

'CONST' KANAL WEJSCIOWY = 1, KANAL WYJSCIOWY = 6.

‡           KODY ZNAKOW WEDLUG ISO-7
'CONST' SP = 32, EOT = 4.

'ACTION' WCZYTAJ ZNAK +ZNAK) =
            "$L    LWT,1," KANAL WEJSCIOWY
            "$L    EXT C2IO,INCHAR"
            "$L    RJD,5,INCHAR"
            "$L    RWD,2," ZNAK .

'ACTION' WYPISZ ZNAK +)ZNAK =
            "$L    LWT,1," KANAL WYJSCIOWY

            "$L    LWI,2," ZNAK
            "$L    EXT C2IO,OUTCHAR"
            "$L    RJD,5,OUTCHAR" .

'ACTION' PRZYLACZ KANALY WE WY =
            "$L    EXT C2SI01,C2SO06"
            "$L    RJD,5,C2SI01"
            "$L    RJD,5,C2SO06" .

'PRELUDE' PRZYLACZ KANALY WE WY.

'ENDSEC' PODSTAWOWE WEJSCIE WYJSCIE NA MERZE 400.

'ENDLAY' L.

'PRELUDE' PODSTAWOWE WEJSCIE WYJSCIE NA MERZE 400.

'ENDMOD' PODSTAWOWE OPERACJE NA MERZE 400.
```

Rys. 3

Przy rozwijaniu makrodefinicji teksty w cudzysłowach (") są kopiowane literalnie, zaś nazwy zastępowane są wyrażeniami adresowymi stosownie do rodzaju obiektów będących odpowiednimi parametrami aktualnymi wywołania. Kompilator rozwija makrodefinicje jako tekst, bez dodatkowej interpretacji, stąd podział na linie (instrukcje asemblera) musi być dokonany explicite przez programistę. Sekwencja SL odpowiada znakowi CR.

Godną uwagi jest definicja „zaniedbaj”. Jej rozwinięciem w programie docelowym jest linia komentarza. Celowość definiowania reguły nie mającej wpływu na semantykę programu wynikowego zostanie uzasadniona później.

Oczywiście akcje w sekcji „podstawowe wejście wyjście...” są odwołaniami do podprogramów bibliotecznych. Biblioteka ta jest obowiązującym standardem tylko dla programu będącego kompilatorem CDL2, choć nadaje się do większości zastosowań. Nie ma powodu, żeby dla specjalnych potrzeb nie implementować innego zestawu operacji lub nie zrobić tego w inny sposób.

Korzystając z modułu „podstawowe operacje...” uzupełnimy teraz przykład do kompletnego programu. Drugi moduł — definiujący program użytkowy — nie zawiera już elementów zależnych od maszyny. Rzeczywiste środowisko zastępuje w nim sekcja importująca potrzebne algorytmy (rys. 4).

W przykładach widać również specyfikację afiksów zależnie od sposobu ich przekazywania: dziedziczone +>x (przez wartość), generowane +x> (przez wynik), przechodnie +>x> (przez wartość/wynik).

Oprócz ewentualnych afiksów generowanych i przechodnich oraz modyfikacji zmiennych (efektów ubocznych) każdy algorytm przekazuje jeszcze jeden wynik, będący jedną z dwu wartości: sukcesem lub porażką. Sukces i porażka są wykorzystywane do konstrukcji struktur sterowania w CDL2. O ile makrodefinicje zawierają zwykle proste sekwencje instrukcji maszynowych (bez żadnych rozgałęzień i pętli), o tyle reguły zapisane w CDL2 zawierają wyłącznie wywołania algorytmów (reguł bądź makro). Spójnikami programotwórczymi są: przecinek (,) — oznaczający następstwo wywołań i średnik (;) — będący separatorem prostych ciągów wywołań zwanych alternatywami. Obliczenie alternatywy polega na kolejnym wywołaniu jej elementów aż do końca (;) lub do wystąpienia pierwszej porażki. Napotkanie końca oznacza sukces alternatywy, w przeciwnym razie mamy do czynienia z jej porażką. W razie porażki dalsze obliczanie alternatywy zostaje zaniechane i podejmuje się próbę obliczenia następnej alternatywy w regule. Sukces alternatywy jest sukcesem reguły, porażka wszystkich alternatyw — jej porażką. Elementarnym sukcesem jest wywołanie oznaczone przez plus (+), elementarną porażką — minus (—). Plus i minus dotyczą całej reguły. Sukces i porażka mają także swoją realizację w modelu wykonawczym (ang. *run-time model)* makrodefinicji.

```
'MODULE' KOPIOWANIE Z KOMPRESJA SPACJI.

'LAYER' L.

'SECTION' RZECZYWISTE SRODOWISKO.

'EXT' WCZYTAJ ZNAK, WYPISZ ZNAK, SP, EOT, JESLI ROWNE, USTAL.

'IMPORT' WCZYTAJ ZNAK, WYPISZ ZNAK, SP, EOT, JESLI ROWNE, USTAL.

'CONST' SP, EOT.

'FUNCTION' USTAL +X) +)Y .

'TEST' JESLI ROWNE +)X +)Y .

'ACTION' WCZYTAJ ZNAK +ZNAK) .

'ACTION' WYPISZ ZNAK +)ZNAK .

'ENDSEC' RZECZYWISTE SRODOWISKO.

'SECTION' KOPIOWANIE.

'INV' WCZYTAJ ZNAK, WYPISZ ZNAK, SP, EOT, JESLI ROWNE, USTAL.

'ACTION' KOPIUJ Z KOMPRESJA SPACJI :
            JESLI NIE KONIEC PLIKU, KOPIUJ NASTEPNY ZNAK,
               KOPIUJ Z KOMPRESJA SPACJI;

'TEST' JESLI NIE KONIEC PLIKU :
            JESLI ROWNE +TRYB +KONIEC, -;


'VAR' TRYB.

'CONST' KOPIOWANIE =0, POMIJANIE =1, KONIEC =2.

'ACTION' KOPIUJ NASTEPNY ZNAK -ZNAK :
            WCZYTAJ ZNAK +ZNAK, WYPISZ +ZNAK,
               USTAL TRYB +ZNAK.

'ACTION' WYPISZ +)ZNAK :
            JESLI ROWNE +TRYB +POMIJANIE, JESLI ROWNE +ZNAK +SP;
            WYPISZ ZNAK +ZNAK.

'ACTION' USTAL TRYB +)ZNAK :
            JESLI ROWNE +ZNAK +SP, USTAL +TRYB +POMIJANIE;
            JESLI ROWNE +ZNAK +EOT, USTAL +TRYB +KONIEC;
            USTAL +TRYB +KOPIOWANIE.

'PRELUDE' USTAL +TRYB +KOPIOWANIE.

'ROOT' KOPIUJ Z KOMPRESJA SPACJI.

'ENDSEC' KOPIOWANIE.

'ENDLAY' L.

'PRELUDE' KOPIOWANIE.

'ROOT' KOPIOWANIE.

'ENDMOD' KOPIOWANIE Z KOMPRESJA SPACJI.
```

Rys. 4

W ten sposób łatwo zinterpretować akcję „kopiuj z kompresją spacji” jako schemat while, akcję „kopiuj następny znak” jako proste następstwo wywołań, akcję „ustal tryb” jako schemat case z klauzulą domyślną itd. Akcja „wypisz” w przypadku sukcesu obu porównań jest pusta, co odpowiada pominięciu kolejnych spacji przy kopiowaniu. Zapewne powinna ona zostać nazwana „ewentualnie wypisz”, aby wyraźnie zasugerować czytającemu możliwość pustego efektu.

Co do części algorytmów wiadomo z góry, że nigdy nie dadzą porażki. Intencję tę programista wyraża specyfikując typ każdego algorytmu. Deklaracja ACTION lub FUNCTION oznacza, że algorytm zawsze kończy się sukcesem. W stosunku do makrodefinicji kompilator przyjmuje tę deklarację „na wiarę”. W stosunku do reguł deklaracja typu jest redundantna i służy zabezpieczeniu przed pomyłkami, umożliwiając sprawdzenie zgodności intencji ze stanem faktycznym.

Druga z kolei klasyfikacja typów algorytmów to rozróżnienie pomiędzy tymi, które mogą mieć efekty uboczne (ACTION, PREDICATE) a tymi, które ich mieć nie mogą

(FUNCTION, TEST). Styl programowania przyjęty w CDL2 głosi, że algorytm kończący się porażką nie może spowodować efektów ubocznych — a kompilator gwarantuje, że zasada ta jest przestrzegana. Z kolei specyfikacja rodzajów afiksów pozwala kompilatorowi dokonać analizy przepływu danych i zasygnalizować wiele błędów. Przykład:

```
FUNCTION dziel przez dziesięć +>x> — szmelc:
  podziel +x + dziesięć +x +szmelc.
CONST dziesięć = 10.
```

powinien być raczej zapisany:

```
FUNCTION dziel przez dziesięć +>x> — reszta:
  podziel +x + dziesięć +x +reszta,
    zaniedbaj +reszta.
```

zwłaszcza że nie ma to wpływu na efektywność kodu wynikowego. W tak małym przykładzie wydaje się to trywialne, ale jest pewne, że gdy po dokonaniu drobnej zmiany w programie liczącym wiele tysięcy linii i bezbłędnie od lat działającym kompilator wydrukuje ostrzeżenie „w regule *xxxx* w linii *nnn* obliczona wartość *zz* nie ma zastosowania", to warto jest poświęcić kilka minut, aby to ostrzeżenie wyjaśnić!

CDL2 NA MERZE 400

Począwszy od 8 wersji kompilatora CDL2 został on wyraźnie podzielony na część niezależną od maszyny docelowej oraz na generator kodu. Korzystając z CDL2 można zatem produkować programy dla tych maszyn, dla których jest dostępny odpowiedni generator kodu. Drugim warunkiem jest stworzenie na maszynie docelowej minimalnego środowiska, głównie zawierającego podstawową bibliotekę wejścia/wyjścia. Program realizujący to środowisko na MERZE 400 zajmuje około 1 K słów pamięci.

Komplet oprogramowania dla MERY 400 został opracowany przez autora w Instytucie Informatyki UW. Jako maszyna źródłowa może występować R-32 lub IBM 360/370 pod nadzorem systemu operacyjnego OS. Kompilator produkuje kod w makroasemblerze SOM 3.

W celach testowych na MERĘ 400 przeniesiono z PDP-11 program będący translatorem /interpreterem/ edytorem języka programowania o złożoności zbliżonej do ALGOL 60. Kod tworzony przez CDL2 z ok. 3K linii źródłowych zajął ok. 20K słów pamięci (bez obszaru na struktury dynamiczne).

Implementacja zawiera podstawowe narzędzia do wspomagania uruchamiania i dokumentacji programów (śledzenie wykonania programu, profil dynamiczny, drukowanie słowników symboli).

Kompilator CDL2 jest również napisany w CDL2, co umożliwia całkowite przeniesienie i instalację tego języka na maszynie MERA 400. Wyeliminuje to największą wadę prezentowanego schematu, jaką jest konieczność fizycznego transportu plików pomiędzy R-32 a MERĄ 400. Dopóki nie przestanie to być problemem, najbardziej celowe jest przenoszenie na MERĘ 400 programów już uruchomionych na R-32.

W trakcie przygotowywania niniejszego artykułu do druku prace nad instalacją CDL2 na MERZE 400, jako maszynie źródłowej, zostały już rozpoczęte i poważnie zaawansowane.

LITERATURA

[1] Bedö A.: Experiences in implementing the ANSWER operating system in CDL2. W: Software-Entwicklung im CDL-Labor. Institut für Software — Technologie Bericht Nr. 45. Gesellschaft für Mathematik und Datenverarbeitung mbH, Bonn 1980

[2] Bolgar G., Langer T., Molnar K.: CDL2 kodgenerator az R10 szamologepre. Szamaki, Budapeszt 1978

[3] Irsigler R.M.A.: Software für ein Rechnernetzwerk in CDL2. Institut für Software-Technologie Bericht Nr. 45. Gesellschaft für Mathematik und Datenverarbeitung mbH, Bonn 1980

[4] Kleine K.: Code generation for CDL2 on small machines. Institut für Software-Technologie Bericht Nr. 45. Gesellschaft für Mathematik und Datenverarbeitung mbH, Bonn 1980

[5] Koster C.H.A.: Using the CDL Compiler-Compiler. W: Compiler Construction, and Advanced Course, Lecture Notes in Computer Science 21, Springer Verlag, Berlin 1976

[6] Koster C.H.A.: CDL — A Compiler Implementation Language. W: Methods of Algorithmic Language Implementation, Lecture Notes in Computer Science 47, Springer Verlag, Berlin 1977

[7] Loeper H., Otter W.: CDL4000 — Ein Rationalisierungsmittel in der Systemprogrammierung. W: Rechentechnik/Datenverarbeitung 1/1979, str. 15—18.

[8] Stahl H. M.: Portability and Efficiency in an Open-Ended Language. Angewandte Informatik 5/79, str. 219—225.

KONFERENCJE

XI Sympozjum Sekcji Cybernetyki Medycznej TIP

Instytut Kardiologii Akademii Medycznej w Poznaniu oraz Środowiskowy Ośrodek Informatyki Politechniki Poznańskiej organizują **XI Sympozjum Sekcji Cybernetyki Medycznej Towarzystwa Internistów Polskich — 23—24 października 1982 r. w Poznaniu.**

Jako tematy wiodące przyjęto:

- **projektowanie i użytkowanie komputerowych baz danych w medycynie**
- **metody i narzędzia informatyki w kardiologii klinicznej.**

Trzon programowy Sympozjum stanowić będą wykłady i referaty przygotowane przez zapraszanych imiennie specjalistów krajowych i zagranicznych. Planuje się także zorganizowanie sesji, na których zostaną przedstawione zgodne tematycznie prace, zgłoszone przez autorów i zakwalifikowane do prezentacji przez Komitet Programowy Sympozjum. Skala tego składnika programu uzależniona będzie od liczby i poziomu nadesłanych prac. Ostateczne decyzje podejmie Komitet Programowy, powoływany obecnie przez organizatorów. Zakłada się, że zakwalifikowane prace będą prezentowane w formie plakatowej (tzw. postery).

W pierwszym dniu Sympozjum zostaną praktycznie zademonstrowane komputerowe systemy gromadzenia i przetwarzania informacji medycznej, opracowane w środowisku poznańskim. Podjęto także starania o przygotowanie na Sympozjum pełnych tekstów prac programowych w formie tomu wydanego przez PWN.

Organizatorzy Sympozjum proszą uprzejmie:

- osoby, które zainteresowane są uczestnictwem w Sympozjum (zwłaszcza te, które dotychczas nie uczestniczyły w Sympozjach Sekcji Cybernetyki Medycznej TIP) — o przesłanie danych personalnych
- autorów, którzy pragną zgłosić prace — o przesłanie roboczych tekstów (3—5 stron maszynopisu, w trzech egzemplarzach) do dnia 31 stycznia 1982 r. O decyzjach Komitetu Programowego autorzy zostaną powiadomieni w kwietniu 1982 r.; ponadto do autorów prac zakwalifikowanych przesłane zostaną ogólne zalecenia dotyczące ostatecznej formy przygotowania pracy.

Korespondencję związaną z Sympozjum należy kierować pod adresem:

Instytut Kardiologii
ul. Długa 1/2
61-848 Poznań
z dopiskiem na kopercie:
XI Sympozjum Sekcji Cybernetyki Medycznej TIP

EDWARD BIELENINIK
Centrum Obliczeniowe
Politechnika Wrocławska

Nowe zastosowania teleinformatyki

O burzliwym rozwoju teleinformatyki świadczą liczne systemy przetwarzania zdecentralizowanego i sieci komputerowe. Są one szczególnie korzystne w przedsiębiorstwach geograficznie rozproszonych, jak np. banki czy magazyny. Informatyka uzupełniona możliwością przesyłania danych na odległość zapewnia całkowitą kompleksowość usług w zakresie przetwarzania danych. Pozwala to na rozwój usług dla bardzo szerokiego kręgu odbiorców, w szczególności usług powszechnych dla społeczeństwa, a więc nie tylko dla użytkowników profesjonalnych. Sprzyja temu rosnące zapotrzebowanie na informację. Rozwój nowych zastosowań teleinformatyki zostanie omówiony poniżej na podstawie zastosowań francuskich.

W zakresie technik przesyłania obserwuje się rozwój uniwersalnych sieci transmisji danych, które mogą być wykorzystywane zupełnie niezależnie przez różne instytucje lub osoby prywatne. We Francji przykładem takiej sieci jest wdrożona pod koniec 1978 r. sieć TRANSPAC, w której zastosowano metodę komutacji pakietów. Dwa lata później przyłączonych było do niej ponad 2000 terminali. Sieć ta jest w dalszym ciągu rozbudowywana.

Równolegle z rozwojem sieci obserwuje się intensywny rozwój systemów dystrybucji informacji graficznej. Jest to dziedzina szczególnie interesująca dla zastosowań powszechnych. Dla uściślenia pojęć w dalszym ciągu, mówiąc o przesyłaniu informacji graficznej, będziemy rozumieli przesyłanie obrazów stałych (zdjęcia i inne dokumenty). Powszechnie stosowana technika nosi nazwę s y m i l o g r a f i i. Polega ona (nie wnikając w techniczne szczegóły poszczególnych rozwiązań) na analizie obrazu wiersz po wierszu, odpowiednim zakodowaniu danych i ich przesyłaniu. Główną cechą tej techniki jest to, że zarówno w odbiorniku, jak i w nadajniku podstawowym nośnikiem informacji jest papier.

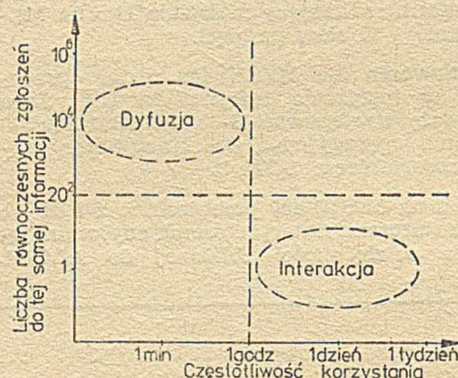
W wielu zastosowaniach interesujące wydaje się zastąpienie nośnika papierowego przez ekran monitora lub odbiornika telewizyjnego. Jest to szczególnie atrakcyjne w dobie powszechnego rozwoju techniki telewizyjnej i nowych metod przesyłania danych cyfrowych. Dlatego też w ostatnich latach podjęto w wielu krajach intensywne prace mające na celu upowszechnienie technik przesyłania informacji graficznej i jej wyświetlania na ekranie. Technikę tę określamy wspólną nazwą wideografii (franc. *videographie)*. Szczególnym przypadkiem systemu przesyłania informacji graficznej jest telerysunek (franc. *telecriture)*. Polega on na wyświetlaniu rysunku — w trakcie jego rysowania — na ekranie, z ewentualną możliwością przesłania na odległość. Poniżej zostaną przedstawione zasady wideografii oraz telerysunku, a także przykłady ich praktycznego zastosowania.



Mgr inż. EDWARD BIELENINIK ukończył w 1970 r. studia na Wydziale Elektroniki Politechniki Wrocławskiej. Pracę zawodową rozpoczął w Instytucie Telekomunikacji i Akustyki Politechniki Wrocławskiej. W latach 1972—1979 pracował w Zakładzie Elektronicznej Techniki Obliczeniowej we Wrocławiu, gdzie zajmował się problematyką systemów operacyjnych. Od 1980 roku pracuje w Centrum Obliczeniowym Politechniki Wrocławskiej, zajmując się zagadnieniami sieci komputerowych, a w szczególności — podsieci transmisji danych. W 1980 r. odbył trzymiesięczny staż we Francji w zakresie teleinformatyki.

WIDEOGRAFIA

Wideografia zajmuje się problemami kodowania i wyświetlania informacji alfanumerycznej lub graficznej na ekranie monitora (odbiornika telewizyjnego). Podstawową jednostką informacji z punktu widzenia użytkownika jest strona. Stanowi ona zbiór informacji wyświetlanej równocześnie na ekranie. Strony są ponumerowane i można je kolejno „listować". Zbiór stron tworzących zamkniętą całość tematyczną tworzy tzw. magazyn.



Rys. 1. Podział systemów dystrybucji informacji

Z punktu widzenia sposobu dystrybucji informacji rozróżnia się wideografię dyfuzyjną (franc. *diffusée*) — TELETEX) i interakcyjną — VIDEOTEX. Zastosowanie pierwszego lub drugiego systemu dystrybucji jest uzależnione od liczby równoczesnych zgłoszeń do tej samej informacji oraz częstotliwości korzystania z systemu. Ilustruje to rys. 1. Oba systemy wideografii są kompatybilne i komplementarne. Ich charakterystyka jest następująca:

- **wideografia dyfuzyjna**
— dystrybucja jest jednokierunkowa
— informacje są krótkie, periodycznie aktualizowane
— informacje są dostępne stale dla szerokiej publiczności

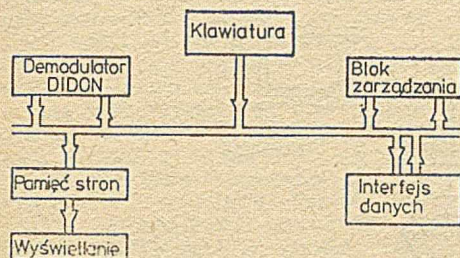
- **wideografia interakcyjna**
— informacje mają dużą objętość (wiele stron)
— informacje są wykorzystywane przez niewielką liczbę abonentów
— dystrybucja odbywa się poprzez telefoniczną sieć komutowaną.

W pierwszym przypadku użytkownik wybiera rodzaj usługi (magazyn), wykorzystując specjalną klawiaturę, w drugim przypadku — klawiaturę i aparat telefoniczny.

Kodowanie

Stosowane są różne systemy kodowania. Obok trybu alfanumerycznego (alfabet CCITT nr 5 — 128 znaków 7-bitowych) wykorzystuje się tryb graficzny (system mozaikowy, geometryczny, fotograficzny itp.). We Francji opracowano system kodowania i wyświetlania informacji wi-

deograficznej ANTIOPE (Acquisition Numerique et Televisualisation d'Images Organisées en Pages d'Ecriture). Zaadaptowano 7-pozycyjny kod ASCII, uzupełniony zestawem znaków semigraficznych i sterujących. Każdy podstawowy element ekranu jest dzielony na sześć elementarnych, wyświetlanych niezależnie prostokątów. Ekran dzieli się zatem na 40 × 24 × 6 = 5760 elementarnych prostokątów. Główne funkcje znaków sterujących to określenie koloru tła, zmiana szerokości (lub wysokości) znaków, wybór pomiędzy zestawem znaków alfanumerycznych i graficznych itp. W systemie można wykorzystywać po 16 różnych alfabetów. Poza tym system ANTIOPE charakteryzuje się niezależnością funkcji kodowania i przesyłania, łatwym dostosowaniem do różnych standardów systemów telewizyjnych oraz bogatym repertuarem znaków.



Rys. 2. Podział blokowy terminala systemu ANTIOPE

Schemat blokowy terminala systemu ANTIOPE przedstawiono na rys. 2. Jego główne funkcje to wybór danych z sygnału wizyjnego, wyświetlanie na ekranie i ewentualnie — wykorzystanie w inny sposób odebranych danych (np. drukowanie). Za pomocą bloku sprzęgającego można odbierać dane z innej sieci (np. telefonicznej).

Organizacja magazynu stron

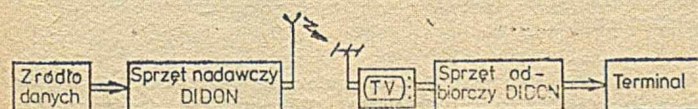
Jak już wyżej wspomniano magazyn informacyjny składa się z określonej liczby stron tworzących jednostkę tematyczną. Każda strona zawiera 24 wiersze 40-znakowych oraz jeden wiersz nagłówkowy, zawierający informacje organizacyjne, który — w zasadzie — nie jest wyświetlany na ekranie. Średni czas dostępu do strony zależy od liczby stron w magazynie oraz liczby wykorzystywanych do przesyłania linii (kanałów).

Zastosowanie

Praktyczne stosowanie systemu ANTIOPE zapoczątkowano w 1977 r. (magazyn giełdowy ANTIOPE-bourse). Od stycznia 1979 r. uruchomiono magazyn meteorologiczny (ANTIOPE-meteo), dostarczający danych meteorologicznych zarówno dla szerokiej publiczności, jak i dla wąskiej grupy odbiorców profesjonalnych. Przewiduje się bardzo intensywny rozwój zastosowań powszechnych (np. telegazeta, ogłoszenia, gry, rezerwacje) i profesjonalnych (np. banki danych, obrót magazynowy, telekurier, telerysunek).

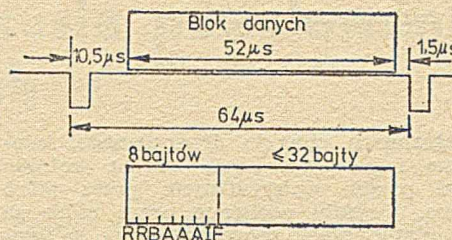
TELETEX

Schemat ogólny systemu przedstawiono na rys. 3. Jego głównym elementem jest moduł DIDON (Diffusion de DONnées). Moduł ten realizuje następujące funkcje:
- tworzenie pakietów danych (32 bajty danych + 8 bajtów organizacyjnych)
- modulacja NRZ
- modulacja danych i sygnału wizyjnego (po stronie nadajnika)



Rys. 3. Schemat ogólny systemu dystrybucji dyfuzyjnej

- demodulacja danych i sygnału wizyjnego (po stronie odbiornika)
- multipleksowanie (demultipleksowanie) sygnału informacyjnego i wizyjnego.



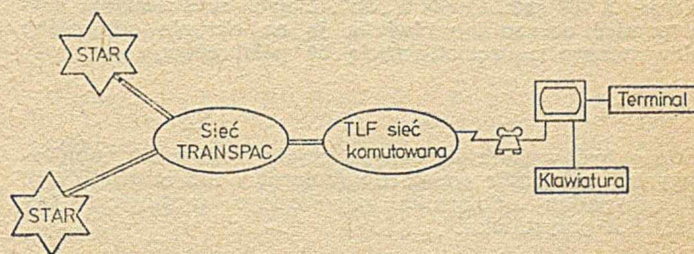
Rys. 4. Przesyłanie danych w systemie DIDON

Sygnał informacyjny przesyła się w sygnale wizyjnym, wykorzystując bądź cały kanał, bądź sygnał powrotny wygaszania ramki (rys. 4). Typowe dane charakterystyczne systemu wideografii dyfuzyjnej, wykorzystującego standard telewizyjny L-SECAM, zestawiono poniżej:

— częstotliwość ramka/obraz — 50/25
— liczba linii — 625
— wielkość pakietu (bajty) — 8 + 32
— szybkość przy 1 linii/ramkę (bit/s) — 12 800
— szybkość przy wykorzystaniu pełnego kanału (Mbit/s) — 4
— liczba bajtów informacyjnych na linię — 32
— liczba znaków w wierszu — 40
— liczba wierszy na stronę — 25
— liczba stron/s przy 1 linii/ramkę — 2
— maksymalna pojemność programu (str/s) — 14
— pojemność maksymalna pełnego kanału (295 linii) — 590 str/s.

VIDEOTEX

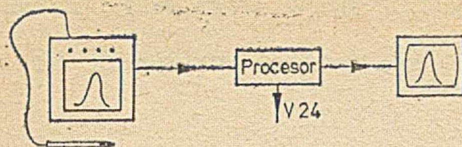
W systemie interakcyjnym informacje są przesyłane na żądanie zgłaszane poprzez telefoniczną sieć komutowaną (rys. 5). Dostęp do poszczególnych rodzajów (magazynów) informacji jest chroniony systemem kluczy. Tylko abonenci posiadający odpowiedni klucz mają dostęp do chronionej informacji. Opracowany we Francji system nosi nazwę TELETEL. Dostęp abonenta do bazy danych wideograficznych jest organizowany w tym przypadku za pomocą systemu STAR (Source Teletel Acces Reseau).



Rys. 5. Schemat systemu dystrybucji interakcyjnej

TELERYSUNEK

Obok wideografii dyfuzyjnej i interakcyjnej, bardzo interesujący, szczególnie dla zastosowań profesjonalnych, wydaje się być system telerysunku, umożliwiający przetwarzanie rysunku w trakcie jego rysowania, celem wyświetlenia na ekranie lokalnego lub zdalnego monitora (odbiornika telewizyjnego). Schemat ogólny systemu przedstawiono na rys. 6. Poniżej zostaną scharakteryzowane funkcje poszczególnych jego modułów.



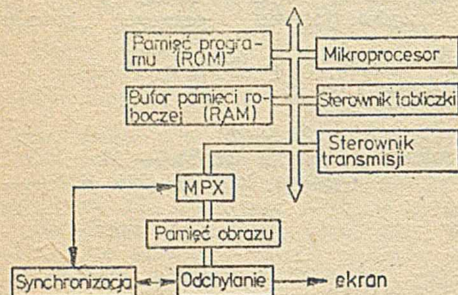
Rys. 6. Schemat ogólny systemu telerysunek

Tabliczka

Składa się z dwustronnego obwodu drukowanego, tworzącego linie poziome i pionowe. Linie te są kolejno oświetlane impulsami prądu, wykrywanymi przez pióro będące swoistą anteną. Sygnały odebrane przez pióro są wzmacniane i przetwarzane przez procesory celem odtworzenia współrzędnych pióra na tabliczce. Tabliczka jest wyposażona w cztery przełączniki umożliwiające m.in. zmianę koloru, wycieranie rysunku, kasowanie ekranu. W obecnej wersji systemu wykorzystuje się dwa kolory (czerwony i zielony).

Procesor

Składa się z mikroprocesora, pamięci programu, sterowników, pamięci obrazu i obwodu wyświetlania. Schemat blokowy procesora przedstawiono na rys. 7. Jego funkcją jest odbiór, przetwarzanie, kasowanie i wyświetlanie rysunku.



Rys. 7. Schemat blokowy procesora systemu telerysunek

Podstawową funkcją mikroprocesora jest sterowanie wejściem i wyjściem procesora graficznego oraz zarządzanie systemem. Sterowniki realizują niezbędną konwersję sygnału podczas zdalnej transmisji. Pamięć obrazu składa się z dwu płatów (jeden dla koloru zielonego, drugi — czerwonego). Zastosowano pamiętanie punktowe obrazu, co wymaga stosunkowo dużej pojemności pamięci obrazu (512 wierszy × 768 punktów × 2 bity). Obwód wyświetlania czyta pamięć obrazu i wyświetla ją na ekranie odbiornika telewizyjnego z częstotliwością 50 półobrazów/s.

Odbiornik tv

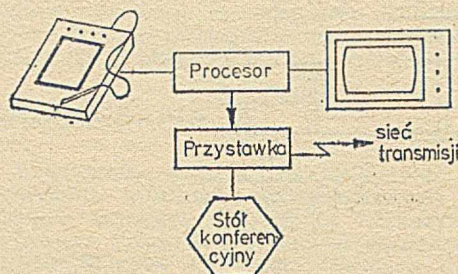
Może być wykorzystywany specjalny odbiornik monitorowy lub odbiornik popularny (kolorowy lub czarno-biały). W przypadku odbiornika popularnego wymagana jest specjalna przystawka.

ZASTOSOWANIA

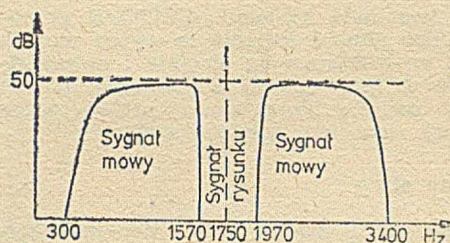
- Przesyłanie programów audiograficznych. Program składa się z rysunku oraz towarzyszącego mu dźwięku (komentarza). Wykorzystując DIDON można w jednym kanale równocześnie przesyłać do 50 programów audiograficznych. Program może być także przesyłany przez nadajniki radiowe AM lub FM.
- Łączność audiograficzna w sieci telefonicznej komutowanej. Rysunek i dźwięk są multipleksowane w jednym łączu telefonicznym i przesyłane równocześnie. Odbywa się to bez istotnego zakłócenia mowy. Typowym praktycznym zastosowaniem tej techniki jest system telekonferencji.
- Rejestracja z odtwarzaniem programów audiograficznych z kasety.
- Mieszanie obrazów telerysunku z innymi obrazami wizyjnymi (napisy na obrazach, efekty specjalne itp.).

Telekonferencje

Telekonferencje stanowią bardzo ciekawy przykład zastosowania systemu telerysunku. Umożliwia on prowadzenie rozmowy i równoczesne przesyłanie rysunku (współrysowanie) przez osoby zgrupowane w połączonych linią telefoniczną studiach konferencyjnych. Schemat blokowy wyposażenia takiego studia przedstawiono na rys. 8. Jest to system telerysunku uzupełniony o stół konferencyjny (wyposażony w mikrofony i głośniki) oraz specjalną przystawkę (multipleksowanie rysunku i mowy oraz modulacja). Rysunek jest przenoszony w paśmie sygnału mowy (rys. 9), co bez istotnego pogorszenia jakości mowy pozwala na ekonomiczne wykorzystanie linii telefonicznej.



Rys. 8. Schemat blokowy wyposażenia studia telekonferencyjnego



Rys. 9. Przenoszenie sygnałów mowy i rysunku w systemie telekonferencji

* * *

Opisane wyżej systemy są już od kilku lat stosowane w codziennej praktyce (wspomniane Antiope-bourse i Antiope-meteo). Eksperymenty z zastosowaniem systemu dystrybucji dyfuzyjnej prowadzone są od połowy 1978 r. w Paryżu i Lyonie. Bardzo aktywne pod tym względem jest Rennes, które w ostatnich latach zyskało miano stolicy teleinformatyki francuskiej [1].

Wersja interakcyjna (VIDEOTEX) systemu wideografii pod nazwą TELETEL jest, poczynając od 1980 r., przedmiotem eksperymentu praktycznego w regionie Velizy pod Paryżem, gdzie w prywatnych mieszkaniach zainstalowano 3000 terminali tego systemu. Głównym celem eksperymentu jest zbadanie aspektów socjologicznych wdrożenia systemu.

Prace w zakresie systemu VIDEOTEX są prowadzone w wielu krajach. Odpowiednikiem francuskiego systemu TELETEL jest w Wielkiej Brytanii — PRESTEL, w Kanadzie — TELIDON, w Japonii — CAPTAIN.

Na koniec warto wspomnieć o ciekawym projekcie pod nazwą „Książka elektroniczna". Projekt przewiduje wprowadzenie ogólnokrajowego systemu automatycznej ewidencji abonentów telefonicznych. Pozwoli to na całkowite wyeliminowanie książek telefonicznych i zautomatyzowanie większości prac związanych z administrowaniem siecią telefoniczną.

---

[1]) Większość ww. prac jest stymulowana przez DGT (Direction Generale de Telecommunications). Głównymi wykonawcami są: CCETT — Centre Commun d'Etudes de Telediffusion et Telecommunications oraz, Cap Gemini Sogeti.

LESZEK A. MACIASZEK
Akademia Ekonomiczna
Wrocław

Słownik-skorowidz w procesie powstawania i utrzymywania baz danych

Projektowanie i rozwój złożonych systemów informatycznych z bazą danych wymaga wykorzystywania słownika-skorowidza danych, rozumianego jako meta-baza danych opisująca bazę użytkową. Ta meta-baza może być oprogramowana indywidualnie z myślą o konkretnym systemie informatycznym (wówczas do jej programowania można wykorzystać ten sam system zarządzania bazą danych, pod nadzorem którego działa baza użytkowa), z zasady jednak powinna być ona zarządzana przez odrębny pakiet słownika-skorowidza danych (ang. *data dictionary/directory* — DD/D). Pakiety te mają najczęściej charakter autonomiczny (np. DATA MANAGER, LEXICON, DATA DICTIONARY), chociaż istnieją też produkty stanowiące rozszerzenie określonego systemu zarządzania bazą danych (np. Zintegrowany Słownik Danych IDMS).

Słownik-skorowidz danych powinien być wykorzystywany zarówno w procesie powstawania (projektowania, programowania i zakładania) bazy danych, jak i w procesie jej utrzymywania (eksploatacji i rozwoju). Pierwszy przypadek utożsamia się z wykonywaniem przez pakiet funkcji słownika, natomiast proces utrzymywania bazy wiąże się raczej z funkcją skorowidza [4(s.5—15); 8(s.53)]. Funkcja słownika realizowana jest przede wszystkim poprzez udostępnianie odpowiednich informacji projektantom i programistom (tzw. bierny słownik-skorowidz). Z kolei funkcja skorowidza wymaga aktywnego i bezpośredniego współdziałania z systemem zarządzania bazą danych (czynny słownik-skorowidz).

Pełna realizacja funkcji słownika-skorowidza danych wymaga ścisłego powiązania go z bazą danych i zintegrowania pakietu słownika-skorowidza z systemem zarządzania bazą danych. Innymi słowy, zarówno słownik-skorowidz, jak i oprogramowanie nim sterujące, powinny zajmować centralną pozycję w stosunku do innych składników systemu informatycznego z bazą danych [2(s.159, rys. 1—3); 9(s.4, rys. 2)]. W tym celu słownik-skorowidz danych powinien operować możliwie szerokim zakresem informacji o systemie realizowanym w technologii bazy danych. Co najmniej powinien on zawierać informacje odnoszące się do następujących poziomów: danych prostych i grupowych, zapisów, zbiorów, bazy danych, schematu bazy danych, podschematów bazy, programów, dokumentów źródłowych i wydawnictw [1(s.464); 4(s.5—16); 5(s.73); 7(s.340—342)].

Dla każdego z wymienionych poziomów należy określić nazwy, symbole, definicje, atrybuty, zakresy wartości, algorytmy, synonimy, homonimy, dane kluczowe, liczby wystąpień itp. Między przedstawionymi poziomami słownika-skorowidza, a także między poszczególnymi pozycjami wewnątrz tych poziomów, powinny zostać zdefiniowane powiązania, niezbędne do wykonywania biernej i czynnej funkcji przez tę meta-bazę w stosunku do bazy użytkowej [7(s.341, rys. 7)].

SŁOWNIK-SKOROWIDZ W PROCESIE POWSTAWANIA BAZY DANYCH

W procesie powstawania bazy danych można wyróżnić etapy projektowania, programowania i zakładania bazy. Decyzje projektowo-programowe, jakie w tym procesie zapadają, mają charakter twórczy, nie zrutynizowany. Słownik-skorowidz może stanowić narzędzie komputerowego wsparcia tych decyzji w tym sensie, że może dostarczać zespołom projektowo-programowym szereg informacji decyzyjnych. Można postawić tezę, że w procesie powstawania jest raczej biernie (funkcja słownika).

Spośród wielu funkcji, które — przynajmniej potencjalnie — powinien spełniać słownik-skorowidz [2(s·158—162); 5(s.69—70); 7(s.337—338); 8(s.41)], w procesie powstawania bazy danych można wykorzystać następujące:

- opis danych i narzucanie norm
- opis powiązań i kontrola spójności logicznej
- generowanie danych testowych i procedur integralności
- generowanie kodu schematu i opisów danych w językach bazowych.

Funkcję opisu danych i narzucania norm należy odnosić co najmniej do uprzednio podanych poziomów informacji (dane, zapisy, zbiory, baza, schemat, podschematy, programy, dokumenty źródłowe, wydawnictwa). Dla każdego z tych poziomów powinno być możliwe wyprowadzenie (wydrukowanie i/lub wyświetlenie) informacji niezbędnych w pracach projektowo-programowych. Ze względu na jednoznaczność nazewnictwa, eliminowanie homonimów i synonimów oraz uwzględnianie ograniczeń syntaktycznych i semantycznych, funkcja ta daje możliwość narzucania norm i standardów projektowych.

Funkcja opisu powiązań i kontroli spójności logicznej umożliwia uzyskiwanie informacji w przekrojach międzypoziomowych oraz sterowanie poprawnością i integralnością struktur bazy danych. Dzięki temu projektanci i programiści mogą uzyskiwać odpowiedzi na pytania typu: „Które programy użytkują daną NAZWISKO?", „Jakie podschematy wchodzą w skład schematu KADRY?", „Z jakich typów zapisów składa się zbiór ZAMOWIENIA?" itp.

O ile omówione wyżej funkcje realizowane są — w mniejszym lub większym zakresie — przez wszystkie pakiety słownika-skorowidza danych[1]), o tyle funkcje następne nie zawsze są dostępne w tych pakietach, mimo że zapotrzebowanie na nie jest duże. Jedną z tych funk-



Dr LESZEK A. MACIASZEK ukończył w 1972 r. studia na Akademii Ekonomicznej we Wrocławiu na kierunku Organizacja Przetwarzania Danych. W 1976 r. obronił pracę doktorską. Za obie prace — magisterską i doktorską otrzymał dwie nagrody ministra NSzWiT. Jest autorem ponad 30 publikacji. Pracuje w Instytucie Informatyki i Zakładach Naukowo-Badawczych AE we Wrocławiu, ostatnio na stanowisku kierownika Pracowni Projektowania Baz Danych.

[1]) Nie należy tych funkcji mylić z funkcją DISPLAY (dostępną między innymi w systemie RODAN). Funkcje opisu danych i powiązań słownika-skorowidza współuczestniczą w procesie tworzenia schematu bazy danych (chociaż mogą być wykorzystywane także po zdefiniowaniu schematu), natomiast funkcja DISPLAY jest tylko inną graficznie formą prezentacji zawartości schematu.

cji jest generowanie danych testowych i procedur integralności. Udostępnienie takiego mechanizmu programistom może znacznie przyspieszyć prace związane z zakładaniem bazy danych oraz może mieć znaczny wpływ na poprawność tej bazy. Wynika to między innymi stąd, że ze względu na złożone struktury baz danych zazwyczaj niemożliwe jest zastosowanie do generowania danych testowych programów pomocniczych systemu operacyjnego (np. IEBGD dla OS/JS). Stwarza to konieczność prowadzenia żmudnych prac programowych, mających na celu przygotowanie danych wejściowych (w tym także danych testowych) dla programów zakładania bazy danych. Natomiast ze względu na jakość i skuteczność programów zakładania, duże znaczenie ma możliwość automatycznego generowania przez pakiet słownika-skorowidza kodu sprawdzania integralności bazy danych.

Niekiedy przed pakietem słownika-skorowidza stawia się także zadanie generowania kodu schematu i opisów danych w językach bazowych. Ze względu na projektotwórczy charakter prac nad schematem nie zawsze celowe jest generowanie pełnego i ostatecznego kodu schematu. Stąd też problem ten należy widzieć raczej w aspekcie wspomagania projektanta w jego decyzjach i rozwiązaniach. Natomiast możliwości wykorzystywania przez programistów funkcji generowania opisów danych w językach bazowych (a także możliwości — sugerowanego niekiedy — generowania zdań języka opisu zadań) należałoby uzależniać od łatwości stosowania w tym celu słownika-skorowidza oraz od wielkości i złożoności tych opisów w poszczególnych programach.

Omówione mechanizmy programowe pakietów słownika-skorowidza są dzisiaj powszechnie dostrzegane i doceniane. Specjaliści stoją na stanowisku niemożliwości zbudowania dużych baz danych (rzędu kilkuset milionów lub miliardów bajtów) bez stosowania tych mechanizmów. Co więcej — uważają oni, że utrzymanie takich baz wymaga automatycznego powiązania wielu funkcji słownika-skorowidza z użytkowaną bazą danych i z jej systemem zarządzania.

SŁOWNIK-SKOROWIDZ W PROCESIE UTRZYMYWANIA BAZY DANYCH

W procesie utrzymywania (eksploatacji i rozwoju) bazy danych zmienia się nie tylko zawartość informacyjna bazy, ale także reorganizowane są jej struktury logiczne i fizyczne. Może to powodować konieczność modyfikacji programów i warunków użytkowania bazy oraz stwarzać ryzyko naruszenia jej integralności. Z tego powodu w procesie utrzymywania bazy danych niezbędne jest uzyskanie wsparcia programowego ze strony słownika-skorowidza. Chodzi tu przede wszystkim o wsparcie czynne w znaczeniu omówionej poprzednio funkcji skorowidza.

W procesie utrzymywania bazy danych można oczekiwać od pakietu słownika-skorowidza realizacji następujących funkcji:

- współpracy z programami użytkowymi
- prowadzenia statystyk użytkowania
- wspierania rozwojowych prac projektowo-programowych.

Funkcja współpracy z programami użytkowymi powinna służyć do sterowania całością dostępu do bazy danych ze strony programów użytkowych, procesorów języków zapytań itp. Mówiąc inaczej, słownik-skorowidz powinien w ramach tej funkcji stwarzać możliwość ustalania prawidłowości odesłań do danych, uprawnień dostępu do tych danych oraz ich lokalizacji. Dzięki temu słownik-skorowidz może być z powodzeniem wykorzystywany jako narzędzie utrzymywania integralności bazy danych i jej ochrony (niezależnie od podobnych funkcji spełnianych przez system zarządzania bazą danych). Przykładowo, słownik-skorowidz powinien dawać możliwość wskazywania wszystkich programów operujących na danych, w których wykryto błędy. Pozwala to unikać błędnych wykonań tych programów, a w konsekwencji zabezpiecza przed kontaminacją błędów i naruszeniem integralności bazy danych.

Funkcja prowadzenia statystyk użytkowania polega na zapisywaniu i przechowywaniu w zbiorach słownika-skorowidza informacji o częstotliwości i sposobie wykorzystywania elementów bazy danych oraz o programach, które działały na tych elementach. Dzięki temu możliwe jest wspomaganie innych funkcji słownika-skorowidza (np. funkcji współpracy z programami użytkowymi), a także prowadzenie analiz statystycznych mających na celu tzw. strojenie systemu [2]).

W procesie utrzymywania bazy danych można także wykorzystać — w zależności od potrzeb i zakresu dokonywanych zmian w bazie — większość funkcji słownika-skorowidza właściwych dla procesu powstawania bazy danych. Funkcje te można określić wspólnym mianem wspierania rozwojowych prac projektowo-programowych i można do nich zaliczyć omówione już następujące funkcje szczegółowe:

- opisu danych i narzucania norm
- opisu powiązań i kontroli spójności logicznej
- generowania procedur integralności
- generowania opisów danych w językach bazowych.

Konieczność czynnego wspierania przez słownik-skorowidz procesów utrzymywania baz danych jest obecnie przyjmowana jako aksjomat. Brak tych narzędzi stawia poważne bariery i granice rozwoju systemów z bazą danych. W dalszym ciągu pozostaje jednak sprawą otwartą problem zintegrowania oraz czynnej współpracy pakietów słownika-skorowidza i systemów zarządzania bazą danych. Większość istniejących obecnie pakietów słownika-skorowidza nie spełnia jednak niektórych z omówionych funkcji.

* * *

Współcześnie istnieje na świecie wiele samodzielnych pakietów słownika-skorowidza danych zapewniających możliwość współpracy z różnymi systemami zarządzania bazą danych. Wydaje się, że najbardziej znane z nich to [2(s.160—161); 6(s.327)]:

- LEXICON (współpracujący z systemami IDMS, IMS i TOTAL)
- DATA DICTIONARY (TOTAL)
- DB/DC DATA DICTIONARY (IMS)
- DATA MANAGER (ADABAS, IDMS, IMS, SYSTEM 2000, TOTAL)
- DATA CATALOQUE (IMS, TOTAL)
- UCC 10 (IMS).

Żaden z istniejących pakietów słownika-skorowidza nie pretenduje do miana standardu światowego. Są to rozwiązania indywidualne niezależnych producentów oprogramowania, takich jak: ARTHUR ANDERSON Co., CINCOM SYSTEMS czy UNIVERSITY COMPUTING COMPANY. Wynika to stąd, że badania nad architekturą słownika-skorowidza nie były nigdy prowadzone lub koordynowane przez duże zespoły międzynarodowe (takie jak np. tworzone przez Komitet CODASYL w problematyce baz danych), ani też nie były propagowane do powszechnego stosowania przez tak wielkiego producenta jak IBM (jak to częściowo występuje w odniesieniu do relacyjnego modelu bazy danych). Wydaje się jednak, że wraz z rozwojem zastosowań pakietów słownika-skorowidza problematyka uniwersalizacji i standaryzacji architektury tych pakietów będzie stawała się przedmiotem zainteresowań coraz większych grup specjalistów [3]).

W tych warunkach należałoby także w naszym kraju rozpocząć intensywne prace w zakresie tej problematyki badawczej z myślą dostarczenia użytkownikom pakietu słownika-skorowidza wspierającego (a niekiedy wręcz

[2]) Warto podkreślić, że funkcja prowadzenia statystyk użytkowania istnieje w większości współczesnych systemów zarządzania bazą danych, ale w znaczeniu biernym. Przykładowo, w systemie RODAN zadania takie spełnia Monitor Eksploatacyjny, a także program RPSTAT (procedura skatalogowana — DBUTMS).

[3]) Już obecnie w ramach British Computer Society powołano Grupę Roboczą ds. Systemów Słownika Danych. Pierwszy raport tego zespołu badawczego ukazał się w marcu 1977 roku [3 (s. 330)].

umożliwiającego) powstawanie i utrzymywanie systemów informatycznych z bazą danych (np. z zastosowaniem systemu RODAN). Jednocześnie należy informować potencjalnych użytkowników baz danych, że żmudne i pracochłonne czynności związane z tworzeniem słownika-skorowidza stanowią warunek sine qua non nie tylko sprawnego funkcjonowania bazy danych w przyszłości, ale także są pożyteczne same przez się, gdyż pozwalają wykryć w danych wprowadzanych do systemu wiele nieprawidłowości terminologicznych i semantycznych.

LITERATURA

[1] Cardenas A. F.: Data Base Management Systems. Boston, Massachusetts: Allyn and Bacon 1979

[2] Fry J. P., Teorey T. J.: Design and performance tools for improving database usability and responsiveness. W: Databases: improving usability and responsiveness. Red. B. Shneiderman. New York, San Francisco, London: Academic Press 1978, s. 151—189

[3] Hainaut J. L.: Some tools for data independence in multilevel data base systems. W: Architecture and models in data base management systems. Red. G. M. Nijssen. Amsterdam, Oxford, New York, North-Holland 1977, s. 187—211

[4] Palmer I.: Data base systems: a practical reference. Massachusetts: Q.E.D. Information Sciences, Inc. Wellesley 1975, 1-1 — D-22s.

[5] Postępy w dziedzinie systemów zarządzania bazą danych — 1977. Europejski Program Badawczy Diebolda 99. Warszawa: Zjednoczenie Informatyki 1979

[6] Staniszkis W.: Skorowidz danych jako narzędzie opisu systemu informatycznego: W: INFOGRYF'80. Przegląd projektowania systemów. Szczecin: TNOiK 1980, s. 325—334

[7] Uhrowczik P. P.: Data Dictionary/Directories. IBM Systems Journal 1973 nr 4, s. 332—350

[8] Wytyczne do długofalowej strategii realizacji baz danych. Europejski Program Badawczy Diebolda 104. Warszawa: Zjednoczenie Informatyki 1979, s. 93

[9] Yormark B.: The ANSI/X3/SPARC/SGDBMS architecture. W: The ANSI/SPARC DBMS model. Red. D. A. Jardine. Amsterdam, New York, Oxford: North-Holland 1977, s. 1—21.

JERZY MATWIEJCZUK
Instytut Automatyki Systemów Energetycznych
Gdańsk

ODRA 1300 jako system pośredniczący w programowaniu mikroprocesorów

Jednym z podstawowych warunków powszechnego stosowania mikroprocesorów jest zapewnienie środków umożliwiających łatwe i efektywne ich programowanie. W praktyce stosowane są szeroko systemy, pośredniczące w procesie uruchamiania programów dla systemów mikroprocesorowych. Realizują one — w pełni lub częściowo — cykl przygotowawczy oprogramowania, począwszy od wstępnej obróbki tekstu programu źródłowego, poprzez translację i generowanie programu wynikowego, aż do fazy właściwego uruchomienia i testowania programu.

Niniejszy artykuł prezentuje system uruchomieniowy opracowany w gdańskim oddziale Instytutu Automatyki Systemów Energetycznych, przeznaczony do obsługi pełnego cyklu przygotowawczego oprogramowania mikroprocesorów typu INTEL 8080, przy użyciu systemów komputerowych serii ODRA 1300. Maszyny cyfrowe tej serii, rozpowszechnione w kraju, spełniają zasadnicze warunki stawiane systemom uruchomieniowym w zakresie wyposażenia sprzętowego, dysponując nadto bogatym oprogramowaniem są potencjalnie atrakcyjnymi narzędziami emulacji systemów mikroprocesorowych.

Oprogramowanie prezentowanego systemu uruchomieniowego obejmuje trzy niezależne moduły: makrogenerator, asembler i emulator. Z formalnego punktu widzenia, moduły te tworzą jeden program, dlatego też możliwe jest automatyczne wykonanie pełnego cyklu, tj. nadanie tekstowi programu postaci akceptowanej przez asembler (w fazie obróbki wstępnej), wygenerowanie kodu wynikowego (w fazie translacji) i bezpośrednie przejście do fazy wykonania przetłumaczonego programu. Automatyzm ten polega nie tylko na mechanicznej realizacji kolejnych stadiów cyklu, lecz przede wszystkim — na możliwości zaprogramowania przebiegu testowego, począwszy od punktu startu, poprzez określone sekwencje śledzenia, aż do punktu zakończenia pracy programu. Niezależnie od tych możliwości, faza wykonania jest również nadzorowana przez operatora, dysponującego listą dyrektyw kontrolno-śledzących.

Moduł MAKROGENERATORA obsługuje wstępną fazę translacji, dokonując — zgodnie z pewnymi ogólnymi regułami — przetworzenia tekstu źródłowego programu, poprzedzającego właściwą translację. Analiza i przetwarzanie tekstu źródłowego sterowane są wyodrębnioną grupą dyrektyw (formalnie stanowiących podzbiór dyrektyw asemblera), które odpowiadają następującym funkcjom makrogeneratora:

- definiowanie makroinstrukcji (dyrektywy *MACRO* i *ENDM)*
- obsługa makrowywołań
- warunkowe generowanie tekstu wyjściowego *(IF, ELSE, ENDIF)*
- nadawanie wartości zmiennym *(SET).*

Dwie pierwsze funkcje składają się na podstawowy mechanizm makrogeneratora, który — najogólniej rzecz biorąc — umożliwia zastępowanie określonych fragmentów tekstu (makrowywołań) przez odpowiadające im inne sekwencje tekstu (makrodefincje). Ten uniwersalny mechanizm służy rozszerzeniu języka programowania; cel ten jest osiągany przy zachowaniu reguł języka podstawowego (a więc bez zmiany translatora), bowiem nowo zdefiniowane (makro) instrukcje — po napotkaniu ich w tekście — zostają rozpoznane jako makrowywołania i zastąpione odpowiednimi sekwencjami instrukcji języka asemblera, zawartymi w makrodefinicjach.

W omawianym systemie makrodefinicje mogą być zorganizowane w postaci biblioteki, która może również zawierać standardowe procedury (podprogramy) w postaci źródłowej. Warto dodać, że procedury mogą być w tej fazie dołączane do programu także za pośrednictwem ma-

krowywołań, które są rozwijane — w zależności od określonych warunków — bądź w pełny tekst podprogramu, bądź też w jego wywołanie.

Makrogenerator dopuszcza wielopoziomową strukturę programu, respektując zarówno hierarchię statyczną (wynikającą z zagnieżdżenia makrodefinicji), jak i dynamiczną (wynikającą z zagnieżdżenia makrowywołań). Makrogenerator przestrzega również reguł dotyczących zasięgu identyfikatorów, tj. nazw makroinstrukcji, parametrów makrowywołań, zmiennych itp., a także dopuszcza stosowanie makroinstrukcji rekurencyjnych, jako szczególnego przypadku zagnieżdżonych makrowywołań.

Warunkowy generator tekstu (sterowany dyrektywami *IF... ELSE... ENDIF*) pomija lub generuje określone partie tekstu programu, w zależności od wartości wyrażenia poprzedzonego dyrektywą *IF*.

Translator wyrażeń wykorzystywany jest do oszacowania wartości wyrażeń używanych np. przy warunkowej generacji tekstu, a także przy nadawaniu wartości zmiennym. Zmienne te (których nazwom przyporządkowywane są wartości wyrażeń za pośrednictwem dyrektyw *SET*) mogą mieć zasięg globalny lub lokalny, tzn. zdefiniowany przez określony poziom makrodefinicji. Zmienne o określonej wartości mogą być następnie użyte jako: elementy innych wyrażeń, parametry makrowywołań (zastępowane przez wartość) czy też argumenty instrukcji języka podstawowego. Translator dopuszcza stosowanie operatorów arytmetycznych, logicznych (w tym relacji), operatorów specjalnych (np. przesunięć), a także nawiasów.

Moduł ASEMBLERA funkcjonuje w sprzężeniu z makrogeneratorem; jak wspomniano — ten ostatni można określić jako translator wstępny, który przekształca złożoną (hierarchiczną) strukturę języka źródłowego na „liniową" postać języka asemblera. Asembler wykorzystuje wspólne z makrogeneratorem narzędzia analizy tekstu oraz wspólne struktury danych. W szczególności korzysta on ze wstępnego analizatora tekstu, rozpoznającego podstawowe elementy syntaktyczne tekstu (nazwy, liczby, łańcuchy znaków i ograniczniki). Ponadto respektuje narzuconą przez makrogenerator hierarchię poziomów, rzutującą na operowanie identyfikatorami, którym w tej fazie translacji nadawana jest wartość (etykiety i symbole stałych).

Podstawową funkcją asemblera jest generowanie kodu wynikowego programu. Każdy wiersz programu źródłowego (odebrany na wyjściu z makrogeneratora), poddany analizie syntaktycznej i rozpoznany jako instrukcja lub dyrektywa asemblera, podlega przetworzeniu na binarną postać rozkazu maszynowego, bądź też powoduje podjęcie akcji sterowanej jedną z następujących dyrektyw:

- *ORG, DS* — zmiana bieżącej wartości licznika instrukcji
- *EQU* — nadawanie stałych wartości symbolom
- *DB, DW* — definiowanie zawartości komórek (bajtów i słów)
- *END* — zakończenie przebiegu asemblera.

Proces translacji realizowany jest w dwóch przebiegach. Pierwszy z nich — w którym wykonywana jest większość funkcji asemblera, poza wyprowadzaniem wyników — służy do określenia wartości etykiet, ze względu na możliwość wystąpienia odwołań do nich w tekście programu jeszcze przed ich zdefiniowaniem. W drugim przebiegu powtarzane są wszystkie funkcje obu faz translacji, łącznie z generowaniem wyników, tj. programu wynikowego na nośniku zewnętrznym (lub w pamięci operacyjnej) oraz tabulogramu kompilacyjnego wraz z listą symboli.

Kolejne wiersze tabulogramu kompilacyjnego odpowiadają poszczególnym instrukcjom źródłowym, stanowiącym wynik obróbki makrogeneratora. W każdym wierszu tekst instrukcji poprzedzony jest specyfikacją, obejmującą adres komórki i jej zawartość (jedno-, dwu- lub trzybajtową), a także ewentualną sygnalizację błędu. Lista symboli zawiera nazwy i wartości etykiet oraz symboli definiowanych dyrektywą *EQU*; opcjonalnie drukowane są również symbole lokalne, których lista umieszczana jest bezpośrednio za rozwinięciem makroinstrukcji, w ramach której zostały one zdefiniowane.

Pomyślne zakończenie fazy translacji programu umożliwia automatyczne przejście do etapu wykonania, który ma na celu przetestowanie programu w warunkach symulujących jego przyszłe środowisko. Przebieg procesu testowania może zostać określony z góry (w fazie translacji), za pośrednictwem grupy parametrów traktowanych jako zmienne globalne, o jednoznacznie przyporządkowanych nazwach. Za ich pomocą można określić:

- punkt startu programu, tj. początkową wartość licznika rozkazów (zmienna *START*)
- początkową wartość wskaźnika stosu (*STACK*)
- przedziały śledzenia, tj. pewną liczbę (max. 10) wyodrębnionych sekwencji instrukcji podlegających monitorowaniu (pary zmiennych *TRAPn* i *EXITn*, gdzie *n* jest numerem przedziału, umożliwiającym identyfikację pary)
- punkt zakończenia pracy programu (*STOP*)
- przyporządkowanie adresów urządzeń użytych w programie (w instrukcjach *IN* i *OUT*) urządzeniom systemu uruchomieniowego (zmienne odpowiadające symbolom urządzeń: *TR, TP, CR, LP* i *TW*).

Ta wyodrębniona grupa zmiennych globalnych podlega ogólnym regułom translacji przyjętym w systemie. Tak więc zmienne te mogą przybierać wartości określone przez użycie ich jako etykiet (co jest wygodne w przypadku parametrów określających miejsce — punkt kontrolny — w programie), bądź też za pośrednictwem dyrektyw *SET* lub *EQU*.

Moduł EMULATORA funkcjonuje w systemie jako moduł obsługi fazy wykonania programu; jego podstawową funkcją jest odwzorowanie — na maszynie pośredniczącej — systemu docelowego, a ponadto — nadzór nad przebiegiem testowania.

Symulowanie właściwości mikroprocesora INTEL 8080 odbywa się poprzez interpretacyjną realizację kolejnych rozkazów programu, dostarczanych na wejściu do emulatora w kodzie wewnętrznym INTEL-a. Program ten zawarty jest w całości w pamięci operacyjnej ODRY, mając do dyspozycji ca 22 K słów pamięci, odpowiadających 64 K bajtom maksymalnej przestrzeni adresowej INTEL-a. Emulator odwzorowuje również rejestry robocze mikroprocesora, licznik rozkazów, słowo stanu, wskaźnik stosu oraz wskaźnik blokady przerwań. Umożliwia także symulowanie operacji wejścia/wyjścia oraz przerwań zewnętrznych.

Podstawowe funkcje emulatora wykonywane są w ramach cyklu rozkazowego, obejmującego:

- pobranie kodu rozkazu z pamięci (adres określony przez bieżący stan licznika rozkazów)
- zdeszyfrowanie rozkazu i pobranie jego argumentów
- wykonanie podprogramu odpowiadającego pobranemu kodowi
- aktualizację licznika rozkazów.

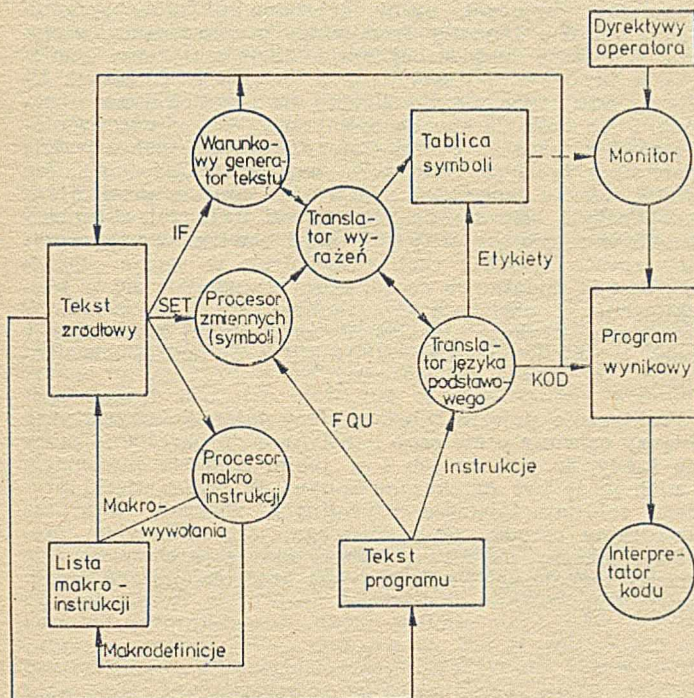
W ramach cyklu rozkazowego odbierane są również przerwania, symulowane przez zgłoszenia operatora. Zgłoszenia te mogą być przyjmowane po zakończeniu realizacji bieżącego rozkazu, a ich obsługa zależy od stanu wskaźnika blokady przerwań (ustawianego przez program). Brak blokady powoduje podjęcie standardowej akcji obsługi przerwania (zapamiętanie na stosie bieżącej wartości licznika rozkazów i wymuszenie restartu programu od adresu określonego przez poziom przerwania), w przypadku blokady — akcja ta jest odkładana do czasu jej programowego zwolnienia.

W każdym cyklu rozkazowym, przed rozpoczęciem wykonywania bieżącej instrukcji, realizowane są funkcje kontrolne emulatora, których zakres zależy od parametrów sterujących przebiegiem testowania. Parametry te mogą być — jak wspomniano — określone w fazie translacji, bądź też w fazie wykonania, w trybie bezpośredniej współpracy z operatorem. Funkcje kontrolne, sterowane tymi parametrami, sprowadzają się głównie do monitorowania wybranych fragmentów programu (przedziałów śledzenia), tj. drukowania stanu rejestrów i wskaźników procesora po wykonaniu instrukcji zawartych w tych przedziałach.

Automatyczna kontrola wykonania programu obejmuje również monitorowanie instrukcji wejścia/wyjścia, za pośrednictwem parametrów przyporządkowujących programowe adresy urządzeń — urządzeniom systemu uruchomieniowego. Urządzenia nie zdefiniowane mogą być symulowane przy użyciu konsoli operatorskiej. Operacje wyjścia monitorowane są poprzez drukowanie adresu urządzenia i numerycznego kodu informacji wyprowadzanej, a operacje wejścia — poprzez drukowanie adresu urządzenia i zawieszanie akcji programu w oczekiwaniu na wprowadzenie informacji przez operatora.

Niezależnie od automatycznego przebiegu fazy testowania, sterowanego odpowiednimi parametrami, emulator dopuszcza możliwość bezpośredniego operatorskiego nadzoru nad realizacją programu (w tym również możliwość określania przez operatora tych właśnie parametrów). Zakres kontroli operatorskiej obejmuje:

- ładowanie programu do pamięci (dyrektywa *LO*)
- wyprowadzanie programu na taśmę papierową (*DU*)
- zawieszanie i uaktywnianie programu (*SU, GO*)
- wyprowadzanie zawartości komórki pamięci lub rejestru (rejestrów) programu (*OU*)
- zmianę zawartości komórki pamięci lub rejestru (*AL*)
- ustawianie i kasowanie trybu monitorowania w określonych przedziałach śledzenia (*TM, NM*)
- definiowanie urządzeń zewnętrznych (*AS*)
- symulowanie operacji wejścia dla urządzeń nie zdefiniowanych (*IN*)
- symulowanie przerwań zewnętrznych (*RS*)
- określanie punktu zakończenia pracy (kasowania) programu (*DE*).



Współdziałanie modułów systemu uruchomieniowego

Zakończenie fazy emulacji następuje bądź automatycznie, po osiągnięciu przez program określonego punktu końcowego, bądź też — w dowolnym momencie — na rozkaz operatora. W obu przypadkach skasowanie programu powoduje ustawienie początkowych warunków dla kolejnego przebiegu i umożliwia powtórzenie fazy emulacji lub realizację nowego, pełnego cyklu uruchomieniowego (rys. 1).

WARUNKI EKSPLOATACJI SYSTEMU

Zaprezentowany system uruchomieniowy może być eksploatowany na dowolnym zestawie komputerowym ODRA 1300 z pamięcią operacyjną minimum 32 K słów, pod nadzorem firmowego programu sterującego (np. EX2M dla ODRY 1325). System dopuszcza daleko idącą swobodę w określaniu konfiguracji sprzętowej. Minimalna konfiguracja obejmuje jednostkę centralną z konsolą operatorską, a maksymalna — dwie jednostki pamięci taśmowej (nie licząc ewentualnych bibliotek makroinstrukcji i procedur), czytnik kart, drukarkę wierszową i (lub) czytnik oraz dziurkarkę taśmy papierowej.

Tryb pracy i zakres funkcji realizowanych przez system może być określony przez użytkownika za pośrednictwem listy dyrektyw, bezpośrednio poprzedzających tekst źródłowy programu. Dyrektywy te mogą zostać wprowadzone z czytnika kart, taśmy lub konsoli operatora. Określają one przede wszystkim: urządzenia wykorzystywane dla wprowadzenia tekstu programu (mogą być różne od wybranych dla wprowadzenia dyrektyw, możliwe jest np. bezpośrednie wejście z taśmy magnetycznej), nośniki (nazwy zbiorów i podzbiorów) — wykorzystywane do przechowywania postaci pośredniej tekstu, bibliotek makroinstrukcji oraz programu wynikowego, a także tryb pracy asemblera (ilość przebiegów, sposób generowania kodu wynikowego, sposób zakończenia pracy itp.) i listę funkcji opcjonalnych. Te ostatnie definiują stan przełączników odpowiedzialnych m.in. za wydruk tabulogramu kompilacyjnego i listy symboli, zapamiętanie parametrów sterujących przebiegiem testowania, wyprowadzenie programu wynikowego na taśmę papierową itp.

Jako standard przyjmuje się, że translacja realizowana jest w dwóch przebiegach, z wykorzystaniem taśmy magnetycznej jako nośnika programu wynikowego, oraz że drukowany jest tabulogram kompilacyjny z listą symboli globalnych. Przechowane też zostają parametry dla fazy testowania, do której przejście następuje bezpośrednio, bez zawieszania programu (chyba, że wystąpiły błędy w fazie translacji).

Listę wymienionych udogodnień eksploatacyjnych rozszerzyć można o możliwość korzystania z oprogramowania firmowego w zakresie edycji tekstów programów, zakładania i aktualizacji bibliotek itp., system bowiem respektuje wymagany przez te programy standard w odniesieniu do formatu zbiorów, wykorzystywanych w trakcie realizacji cyklu uruchomieniowego.

LITERATURA

[1] Brown P. J.: Makrogeneratory i oprogramowanie przenośne. WNT, Warszawa, 1978
[2] Górnicki T., Wyrzykowski M.: Uruchamianie systemów mikroprocesorowych przy użyciu MERY 303. INFORMATYKA nr 7/1979
[3] Hughes T. P., Sawin D. H.: Breakpoint design for debugging microprocessor software. Computer Design, November 1978
[4] INTEL 8080/85 Assembly Language Programming Manual
[5] The Engineering of Microprocessor Systems. Pergamon Press, 1979.

W następnym numerze: rządowy raport o stanie polskiego przemysłu komputerowego, statut Polskiego Towarzystwa Informatycznego, dyskusja na temat Systemu Państwowej Informacji Statystycznej, informacja o Komisjach Koordynacyjno-Porozumiewawczych „Solidarności".

TADEUSZ JAEGERMANN
Zarząd Mechanizacji i Automatyzacji
Opracowań Statystycznych GUS
Warszawa

Ochrona danych obowiązuje

Zajmując się od kilku lat ochroną danych w pionie obliczeniowym Głównego Urzędu Statystycznego coraz wyraźniej widzę wieloaspektowość i złożoność tej problematyki. Przyznaję, że w okresie, gdy organizowałem duży ośrodek elektroniczny, a następnie w pewnym okresie nim kierowałem, zawsze ceniłem zadania ochrony i szanowałem trud specjalistów ukierunkowanych na doskonalenie metod ochrony danych. Ale dopiero teraz zdaję sobie w pełni sprawę z tego, jak wiele trzeba wiedzieć, ile wątków kojarzyć, aby można było uznać, że specjalista koordynujący problematykę ochrony danych dostatecznie spełnia swe zadania.

Wydaje się, że dość często jeszcze rozpowszechnione jest mniemanie, iż w obecnych warunkach eksploatacji sprzętu w Polsce, wykorzystywanego głównie w trybie pracy wsadowej, można niemal nie mówić o potrzebie ochrony danych. Wystarczy — jak niektórzy sądzą — zamykać drzwi do sali komputerowej i w stosowny sposób zorganizować biurowe czynności dostarczania danych oraz kontroli i wysyłki wyników przetwarzania.

Niejednokrotnie spotykałem się z poglądami, że „maniacy" ochrony danych po prostu „zarazili się" podejrzliwością zachodnich ekspertów od przetwarzania danych na potrzeby banków, policji czy wojska. W tych obszarach działalności uwidacznia się bowiem z całą ostrością pierwszoplanowe znaczenie ochrony danych zarówno przed możliwą kradzieżą lub nieprawnym wykorzystaniem tajemnic firm i ich klientów, poszczególnych obywateli czy państwa, jak i konsekwencjami nagłego zahamowania usług komputerowych w wyniku świadomych wrogich działań.

Oczywiście każdy typ ochrony powinien być stopniowany oraz dostosowany do konkretnych warunków, rozpoznanych potrzeb i przewidywanego rzyka. A ponieważ ochrona danych musi kosztować, niezbędna jest więc zawsze analiza projektowanego zakresu zastosowania środków ochronnych oraz szczegółowy rachunek kosztów spodziewanych korzyści.

Podstawową jednak prawdą, która ma zastosowanie w odniesieniu do **k a ż d e g o** ośrodka komputerowego — jest konieczność stałego obserwowania wszelkich zagrożeń, jakie zawsze pojawiają się w elektronicznym przetwarzaniu danych. Występują one zarówno w warunkach technologii konwencjonalnych (przetwarzania lokalnego w trybie wsadowym), jak i w technologiach bardziej zaawansowanych. Zrozumiałe, że problematyka ochrony danych w nowoczesnych systemach przetwarzania rozproszonego, obsługujących ważne dziedziny życia społecznego i gospodarczego, staje się kwestią kluczową, bez rozwiązania której takie przetwarzanie staje się w ogóle nie do pomyślenia. Zawsze powinniśmy jednak pamiętać, że rozwój i doskonalenie informatyki zależy nie tylko od nakładów inwestycyjnych, możliwości instalowania nowego sprzętu, ale również od fachowego poziomu kadr informatyków, od zgromadzonego doświadczenia wiedzy, również w dziedzinie szeroko pojętej ochrony danych.

Przez ochronę danych w niniejszych rozważaniach można — zgodnie z ogólnie przyjętą definicją — traktować zbiór środków zabezpieczających dane przed rozmyślnym lub przypadkowym, nieuprawnionym dostępem, oraz ich modyfikacją lub zniszczeniem. W piśmiennictwie w języku angielskim ujmowane jest to terminem „security". Jest to dość duży obszar tematyczny zagadnień, obejmujący m.in. cechy ochronne oprogramowania systemowego (w tym systemu operacyjnego), cechy ochronne układów sprzętowych, ochronę fizyczną środowiska przetwarzania, wytyczne dotyczące bezpiecznej eksploatacji systemów. Ponieważ w bardzo wielu instytucjach, w tym w Głównym Urzędzie Statystycznym, przetwarza się również dane o osobach i ich gospodarstwach, konieczne jest tu uwzględnianie również aspektu ochrony sfery ich osobistych tajemnic (tzw. sfery życia prywatnego — angielskie *„privacy"*). Ten ostatni termin angielski odpowiada innemu przekrojowi problematyki ochrony danych. Nawet najlepiej chroniony system w sensie „security" nie musi bowiem w wystarczający sposób spełniać cech ochronnych w sensie *„privacy"*. Jest to dodatkowa cecha, którą uwzględnia się w projektowaniu i eksploatacji systemów informatycznych. W niniejszym artykule nie zostanie ona szerzej rozwinięta, z czego jednak nie wynika, iż dziedzinę tę można lekceważyć.

Praktycznie pojęcie ochrony sfery tajemnic prywatnych dotyczy ochrony praw osób i instytucji do określania przez nie kiedy, w jaki sposób i w jakim zakresie informacja o nich może być przekazana innym. W takim ujęciu problem ochrony tajemnic prywatnych znacznie wykracza poza ramy ośrodka obliczeniowego, jest to bowiem problem społeczny, do rozwiązywania którego nie wystarczają metody techniczne. Niezbędna jest kontrola prawna i społeczna [12].

*

Wagę komputerowej ochrony danych dostrzeżono na Zachodzie 10—15 lat temu. W wyniku wielokierunkowych prac badawczych, prowadzonych zarówno przez firmy produkujące sprzęt i oprogramowanie, jak i firmy konsultacyjne oraz wielkich użytkowników — zgromadzono obszerną dokumentację problemu, opracowano szereg skutecznych metod, a także powstała literatura poświęcona wyłącznie problematyce ochrony danych.

Podjęto próby oszacowania strat, jakie gospodarka ponosi w wyniku nadużyć komputerowych. Na przykład, istnieją oceny, że roczne straty gospodarki brytyjskiej w wyniku tego typu nadużyć są równoważne czterem „napadom stulecia" [7]. Mike Comer, wydawca „Computer Fraud and Security Bulletin" ocenia, że wykrywalność tego typu przestępstw kształtuje się na poziomie poniżej 20%. Ze źródeł amerykańskich można się dowiedzieć, że roczne straty w 1976 r. wyniosły w USA 100 mln dolarów, ale równocześnie rosły w tempie ok. 400% rocznie (A. Pantages. The Price of Protection. DATAMATION No. 3/1976, s. 141).



Inż. Tadeusz JAEGERMANN studiował w latach 1937—1939 na Wydziale Elektrycznym Politechniki Warszawskiej. Dyplom inżyniera elektryka uzyskał w 1957 r., po kontynuacji studiów w Wieczorowej Szkole Inżynierskiej w Warszawie. W latach 1947—1949 pracował w Centralnym Urzędzie Planowania i PKPG, a w latach 1959—1963 w Zakładzie Badań Ekonomicznych i Instytucie Planowania. Jeden z organizatorów Centrum Obliczeniowego Komisji Planowania przy RM, jego wicedyrektor w latach 1963—1977. Od 1977 r. główny specjalista w Zarządzie Mechanizacji i Automatyzacji Opracowań Statystycznych GUS. Aktualnie zajmuje się problematyką technologicznej ochrony danych w pionie informatyki GUS.

Dane te wskazują, iż na pewno przekroczony został w krajach gospodarczo rozwiniętych próg możliwości lekceważenia problemu nadużyć komputerowych. Fakt ten rysuje się szczególnie ostro w świetle stwierdzonego już faktu, że zdecydowana większość występujących w tych krajach strat nie jest ujawniana, a przypadki ujawniane nie są całkowicie reprezentatywne. Są podstawy do przypuszczenia, że takie przestępstwa jak kradzież dokumentów wynikowych, niszczenie zbiorów głównych czy bezprawne modyfikacje programów występują częściej niż na to wskazują dostępne informacje. Instytucje nie są bowiem najczęściej skłonne do przyznania się, że zdarzenia takie w ogóle wystąpiły [21, s. 7].

Prezentując oceny strat powstających w wyniku nadużyć należy również uwzględnić istotny wpływ zmian w technologii prac obliczeniowych. Typowymi przestępstwami w pracach wsadowych realizowanych w trybie przetwarzania lokalnego, w latach sześćdziesiątych i na początku lat siedemdziesiątych, były: kradzież i niekontrolowane kopiowanie taśmy magnetycznej, akty sabotażu w stosunku do sprzętu komputerowego, modyfikacje zapisów itp., a więc działania, dla których niezbędny był dostęp fizyczny sprawcy przestępstwa.

Naturalną jest rzeczą, że szerokie rozpowszechnienie się systemów typu interakcyjnego w istotny sposób zmieniło charakter przestępstw [21, s. 9]. Również w naszych warunkach w coraz większym stopniu musimy się liczyć z nadużyciami typu „kradzież informacji" ze zdalnie działających i z reguły mniej kontrolowanych terminali.

Świadomość zagrożeń jest wprawdzie coraz powszechniejsza, ale jak wskazuje praktyka — nie zawsze jest wystarczającym bodźcem do podejmowania odpowiednich kroków dla przeciwdziałania stratom. Ochrona jest bowiem na ogół bardzo kosztowna. Wiadomo np., że podnoszenie niezawodności sprzętu komputerowego może być uzyskane tylko w wyniku użycia odpowiednio wysokich nakładów, w tym na układy kontrolujące błędy. Podnosi to znacznie koszt sprzętu, w czym często nie są zainteresowani ani jego producenci, ani użytkownicy. Uwzględniając konsekwencje społeczne takich postaw, w krajach gospodarczo rozwiniętych pojawia się ważny czynnik stymulujący ochronę, a mianowicie nacisk prawa. Wystarczy zauważyć, że nowe przepisy narzucające normy ochrony danych zmusiły przedsiębiorstwa w RFN do zatrudnienia z tego tytułu ok. 40 tys. pracowników odpowiedzialnych za ochronę danych [14, s. 38].

Wydaje się, że w Polsce również w coraz większym stopniu powinno się wprowadzać standardy i przepisy zapewniające nadanie właściwego priorytetu działaniom zmierzającym do skutecznej ochrony danych. Doprowadzenie do zbyt dużych opóźnień może nas w ostatecznym rachunku bardzo drogo kosztować.

*

Śledząc literaturę traktującą o ośrodkach kontroli wbudowanych w systemy komputerowe można dojść do wniosku, że nawet w krajach najbardziej zaawansowanych w rozwoju informatyzacji nadal istnieje duża luka między zabezpieczeniami, które sprawdzono już w warunkach laboratoryjnych, a zabezpieczeniami praktycznie stosowanymi w obecnie eksploatowanym sprzęcie. Celem, który realizują producenci, jest włączenie takiego zakresu wewnętrznych zabezpieczeń, aby sprzęt i oprogramowanie nie stanowiły najsłabszych ogniw w całym łańcuchu przedsięwzięć dotyczących ochrony [4, s. 227].

Hamulcem dla tego rodzaju działań są jednak koszty ich realizacji oraz stopień świadomości potencjalnych nabywców. Zdaniem znanego amerykańskiego eksperta Donn Parkera z Instytutu Stanfordzkiego: *„Systemy komputerowe są po prostu zbyt duże i skomplikowane oraz w wyniku nacisku nabywców — zbyt elastyczne i łatwe w użyciu, aby zapewnić mogły znacznie większą odporność na infiltrację, chociaż producenci są w stanie warunek ten spełnić. Producenci ci twierdzą jednak, że dostarczane przez nich systemy komputerowe są obecnie wystarczająco bezpieczne w stosunku do faktycznych potrzeb swych klientów."* [15].

Również w odniesieniu do naszych warunków można wysunąć tezę, że polscy informatycy mają pełną świadomość konieczności wywierania odpowiedniego nacisku na producentów sprzętu i oprogramowania. Przemysł bez wyraźnie sformułowanych postulatów projektantów, programistów i pracowników technicznej obsługi sprzętu nie rozwinie działań w kierunku stworzenia wystarczających instrumentów kontroli, zarówno wbudowanych w systemy komputerowe, jak i w postaci odpowiednich narzędzi analizy — zapewniających tworzenie skutecznych metod ochrony danych.

Równolegle z poszukiwaniem nowych metod warto chyba zwrócić uwagę, iż ochrona ta w bardzo dużym stopniu zależy od jakości sprzętu oraz jego starannego wytestowania, co z kolei łączy się ze wspomnianą już sprawą kosztów.

Poszukiwanie bezpiecznej architektury systemów nie jest bynajmniej sprawą prostą i jak wykazują doświadczenia wymaga bardzo pracochłonnych badań. Szczególnie trudne problemy mogą wyniknąć przy włączaniu nowych rozwiązań ochrony do istniejących już systemów. W rozwiązaniu tych kwestii należy wykorzystywać doświadczenia specjalistów z krajów najbardziej w tej problematyce zaawansowanych. Niestety śledzenie polskiego piśmiennictwa z tej dziedziny wywołuje wrażenie, iż niewiele się dotąd w Polsce na tym odcinku robi.

*

Już dość dawno spostrzeżono, iż najbardziej wrażliwym z punktu widzenia ochrony elementem systemu komputerowego jest jego system operacyjny. W pewnym okresie na Zachodzie powstała swoista „nagonka" na producentów sprzętu, których zaczęto oskarżać o niefrasobliwość w budowie systemów operacyjnych. W 1976 r. na sympozjum w Phoenix (USA) przedstawiciel Instytutu Stanfordzkiego sformułował to lapidarnie: *„Większość systemów operacyjnych przypomina ser szwajcarski"* (DATAMATION No 5/ /1976, s. 180—182). Był to okres, gdy firma IBM starała się „łatać" swe systemy operacyjne za pomocą pakietów typu RSS (Resource Security System), które niestety silnie obciążyły komputery, ale w pewnym jednak zakresie kompensowały błędy w systemach operacyjnych, nie projektowanych z uwzględnieniem potrzeb ochrony danych. Dopiero bowiem systemy operacyjne typu VM370 zaczęły spełniać bardziej zaawansowane wymagania ochronne.

Zdaniem cytowanych już autorów publikacji [4] poważnym problemem jest „niepohamowane uprzywilejowanie", jakie przyznano systemom operacyjnym. W trybie pracy części wykonawczej systemu, tzw. programu dyrygenta, zawieszeniu ulega bowiem większość mechanizmów ochronnych. Powoduje to powstanie zagrożeń typu „Konia Trojańskiego".

Pojawienie się na rynku nowego sprzętu, opartego o technologię układów wielkiej skali integracji, bynajmniej nie poprawiło sytuacji. Powstał bowiem problem mikrokodów, które producenci w wyniku silnej walki konkurencyjnej zaczęli stosować w celu uzyskania sprzętu maksymalnie uniwersalnego. Dla producenta rozwiązanie takie jest wprawdzie bardzo opłacalne, ale odbywa się to kosztem zwiększenia podatności urządzenia na zagrożenia zewnętrzne. Wyszkolony przeciwnik w pewnych warunkach może dokonać — nawet zdalnie — zmian mikrokodu. Mikrokod nie może bowiem być tajemnicą dla użytkownika sprzętu, który musi mieć dostęp do szczegółowej dokumentacji systemu oraz możliwość dokładnego sprawdzenia działania mikrokodu (DATAMATION, No 9/1979, s. 70—82), co niestety nie zawsze występuje w praktyce.

Na tym tle widoczna jest istotna rola przepisów, narzucających systemom określony stopień ochrony. Bez tego rodzaju przepisów trudno myśleć o aktywizacji rzeczywistego zainteresowania ochroną danych. Jak dalece złożony jest to problem wystarczy zauważyć, że Amerykanie spodziewali się wydania takich przepisów na przełomie lat 1979/1980. Brak jest jednak dotychczas informacji czy przepisy te zaczęły już obowiązywać.

Zastrzeżenia, jakie wysuwane są w odniesieniu do cech ochronnych sprzętu, mają również zastosowanie do szeroko pojętego oprogramowania systemów, jakkolwiek z wyłączeniem programów aplikacyjnych, opracowywanych przez personel użytkownika.

Wydaje się, że warto szczególnie zwrócić uwagę na kompilatory. Są to jak wiadomo obszerne i złożone programy,

charakteryzujące się szczególnie tym, że o błędzie generacji kodu można się przekonać dopiero po ukończeniu czasochłonnego przebiegu.

Ogólnie biorąc błędy zawarte w oprogramowaniu mogą się ujawnić nawet po bardzo długim okresie czasu, szczególnie jeśli ukryte są w mało używanych jego fragmentach. Specjaliści znają metody omijania tych przeszkód, ale wielką lekkomyślnością jest ignorowanie informacji o błędach bieżąco przekazywanych przez producenta.

Producenci oprogramowania, działając pod presją konkurencji, a niekiedy umownych terminów dostaw, poszukują w pierwszym rzędzie rozwiązań bardziej uniwersalnych (realizujących więcej funkcji), wymagających mniejszych obszarów pamięci i zapewniających szybsze przebiegi. Nie rzadko, niestety, osiągają ten cel usuwając po prostu większość mechanizmów kontroli programowej, która gwarantuje większą odporność na błędy [16, s. 171].

Panuje pogląd, że oprogramowanie dostarczane odpłatnie, a więc silniej uzależnione od mechanizmów konkurencji, gwarantuje na ogół wyższą jakość z punktu widzenia niezawodności działania, a tym samym ochrony danych.

*

Jeśli projektant systemu informatycznego dysponuje niezawodnym sprzętem, dobrym i gruntownie sprawdzonym oprogramowaniem, wówczas może liczyć na uzyskanie niskiego stopnia jego zagrożenia.

Jest to wprawdzie warunek konieczny, ale nie wystarczający dla stworzenia optymalnych warunków efektywnej ochrony danych. Projektowanie ochrony wymaga bardzo dobrej znajomości całokształtu rozwiązań systemu, który może być zagrożony z różnych stron. Często niestety projektanci koncentrują cały swój wysiłek na jednym tylko aspekcie ochrony, pomijając inne, nie mniej istotne. Nie daje to oczywiście spodziewanego efektu, gdyż przysłowiowy łańcuch pęka wtedy w najsłabszym ogniwie.

Projektant powinien mieć głęboką świadomość tego, że jeśli cokolwiek może się stać złego, to na pewno się to stanie (*„Whatever can go wrong, will go wrong”* — tzw. prawo Murphy'ego). Należy więc zawsze dążyć do wbudowywania w system rozsądnych rozwiązań wykrywających błędy, odzyskujących utracone dane, i to nawet w warunkach najbardziej wydawałoby się nieprawdopodobnych okoliczności.

Dlatego też pozytywną cechą projektanta jest — zdaniem Terry Gibbonsa — pesymizm, gdyż pozwala on zwalczać złudzenia, że można od razu uzyskać bezbłędne programy [6, s. 1].

Oczywiście stwierdzenie to nie dotyczy skrajnie prostych programów wykonywanych w trybie wsadowym, charakteryzujących się przebiegami, które łatwo powtórzyć bez większych konsekwencji jeśli chodzi o terminowość dostarczenia wyników oraz koszty. Ale życie już dziś stawia nowe wymagania, i to niejednokrotnie niezbędne dla realizacji zadań na często przestarzałym już sprzęcie. W takich przypadkach występuje konieczność wielkiej mobilizacji projektantów, celem zabezpieczenia najbardziej wrażliwych danych przed zniszczeniem lub przechwyceniem przez osoby niepowołane.

*

Problemem szczególnie trudnym do realizacji jest do dnia dzisiejszego organizowanie tzw. banków (baz) danych. Wielu autorów obiecywało w Polsce osiągnięcie znaczących sukcesów nawet w oparciu o sprzęt krajowy. Wydaje się, że warto tu być nieco bardziej sceptycznym.

W artykule opublikowanym w czasopiśmie DATAMATION w 1977 r. [2] dwu kompetentnych specjalistów przedstawiło (co prawda 4 lata temu!) półoficjalne poglądy amerykańskiej „Federal Communications Commission” (współdziałającej z AFIPS). Podsumowując wynik trzech międzynarodowych konferencji komunikacji komputerowej (International Conference of Computer Communication — ICCC) autorzy cytowanego artykułu doszli m.in. do następujących wniosków:

- występuje rażące niedocenianie trudności występujących podczas tworzenia i eksploatacji dużych baz danych
- nie można stwierdzić istotnego postępu w metodach projektowania i wdrażania systemów informatycznych, które zapewniałyby ekonomiczne tworzenie takich baz.

Tego rodzaju pesymistyczne oceny odnoszą wspomniani autorzy również do następnej dekady.

James Martin w następujący sposób sformułował wymagania [12, s. 38], jakie powinna spełniać baza danych z punktu widzenia ochrony informacji:

- dane muszą być ochronione przed wszelkimi formami fizycznego zniszczenia
- dane muszą być rekonstruowalne, ponieważ w każdej chwili mogą wystąpić okoliczności nieprzewidziane
- dane muszą zapewniać skuteczną ich kontrolę, gdyż jej brak stwarza potencjalne niebezpieczeństwo oszustw
- system ochrony powinien eliminować możliwość fałszerstw i to w takim stopniu, aby nawet bardzo zdolni programiści nie mogli złamać wprowadzonych barier ochronnych
- obecnie nie ma całkowicie pewnego systemu ochrony, ale należy dążyć aby przełamywanie barier ochronnych było bardzo trudne
- system powinien stwarzać warunki weryfikacji, czy działania użytkownika są w pełni legalne
- działania użytkowników powinny być monitorowane, aby w przypadku stwierdzenia działań nielegalnych lub niewłaściwych można było szybko wykryć ich źródło.

Powyższe wymagania nie są łatwe do spełnienia i dlatego koniecznie jest zawsze drobiazgowa analiza konkretnych warunków uwzględniająca specyfikę potrzeb użytkownika.

Wymaga to również uprzedniego przewidzenia maksymalnej liczby przypadków wystąpienia różnych zagrożeń, a także sprawdzenia odpowiednich procedur programowych. Obowiązuje tu szereg ogólnych przestróg, które w hasłowym sformułowaniu brzmią

- jeśli nie jest to konieczne, nie aktualizuj bazy danych za pomocą procedur w trybie „on-line”
- unikaj aktualizacji jednoczesnej
- oszacuj częstotliwość występowania potrzeby tworzenia kopii awaryjnych zapisów, z punktu widzenia kosztów oraz czasu realizacji
- duże bazy danych dziel na segmenty
- wprowadzaj odpowiednio liczne punkty kontrolne.

*

Systemy komputerowe, w skład których wchodzą terminale, wymagają dodatkowych, specyficznych przedsięwzięć ochronnych. Jeśli terminal nie ma wbudowanych rozwiązań ochronnych, to komputer centralny jest stale narażony na wiele rodzajów zagrożeń.

W każdym systemie terminalowym należy więc przewidzieć następujące procedury zabezpieczeń dostępu:

- identyfikację (unikalna nazwa lub symbol, przyporządkowane danemu obiektowi)
- stwierdzenie autentyczności (sprawdzenie czy obiekt jest tym, za który się podaje (ang. *handshaking procedure)*
- upoważnienie dostępu do chronionego zasobu informacji.

Często u nas stosowana procedura identyfikacji za pomocą hasła nie jest jednak bezpieczną formą ochrony danych, a często stać się może najbardziej wrażliwym elementem tej ochrony. Istnieje bowiem zakorzeniona u nas tendencja do przyjmowania krótkich i łatwych do zapamiętania prostych haseł. Pisanie takich haseł na kartkach i lekkomyślne ich udostępnianie najczęściej występuje w

przypadku nieprawidłowych zachowań. Należy bowiem zdawać sobie sprawę z tego, iż większość systemów komputerowych nie dysponuje skutecznym aparatem ochrony zbiorów. W takiej sytuacji nawet średnio wyszkolony operator może bardzo łatwo wyprowadzić na drukarkę pełną listę haseł. Wynika stąd wyraźnie, że systemy prostych haseł absolutnie nie zapewniają właściwej ochrony przed zdeterminowanym przeciwnikiem.

Silnym lecz słabo dotąd u nas wykorzystywanym narzędziem, jest tworzenie kroniki przebiegów komputerowych. Kronika taka powinna być oczywiście również chroniona przed utratą, np. w wyniku awarii systemu operacyjnego. Uznawana jest jednak ona za bardzo skuteczny sposób odstraszania przed próbami przestępczych działań [8, s. 35].

Terminale są znacznie bardziej niż komputery podatne na uszkodzenia i działania zagrażające danym. Pierwszą linią ochrony przed zagrożeniami jest tu kontrola fizycznego dostępu oraz ustalenie odpowiednich procedur uruchamiania poszczególnych terminali.

Wrażliwym, a jednocześnie najbardziej trudnym do zabezpieczenia elementem systemów terminalowych jest użyta sieć łączności, która powinna być szczególnie silnie chroniona przed dostępem osób nie upoważnionych.

Kable powinny być okresowo sprawdzane na odznaki podsłuchu. Specjaliści ostrzegają [18, s. 71], że bardzo trudno jest przeszkodzić intruzowi w znalezieniu punktu podsłuchu. Dlatego dane szczególnie wrażliwe powinny być szyfrowane.

Systemy terminalowe, działające w trybie on-line, są jak dotąd niestety bardzo łatwe do wrogiej penetracji. Praktyka wskazuje, że przeciwnik, który często waha się przed fizyczną kradzieżą dokumentów, często nie ma żadnych zahamowań przed dostępem do zbioru ze zdalnych terminali [21].

Tej listy ostrzeżeń nie należy w naszych warunkach interpretować jako nawoływanie do wycofywania się ze zdalnego przetwarzania danych. Systemów takich nie wolno tylko stosować w sposób bezkrytyczny. Najgorsze są złudzenia, że w systemach chronionych np. prymitywnym hasłem można przetwarzać dane poufne. Wykorzystajmy te systemy dla przekazywania danych neutralnych, całkowicie jawnych, a co najwyżej traktujmy je jako rozwiązania eksperymentalne, przygotowujące grunt pod przyszłe, właściwie chronione systemy.

LITERATURA

[1] Brown A. R., Sampson W. A.: Programm Debugging. London 1976, s. 127—135

[2] Cerf V. G., Curran A.: The Future of Computer Communications. DATAMATION No 6/1977, s. 105—114

[3] Curtice R. M.: Integrity in Data Base Systems. DATAMATION No 5/1977, s. 64—68

[4] Denning D. E., Denning P. J.: Data Security. COMPUTING SURVEYS, No 3/1979 s. 227—249

[5] Geller S. P.: Erasing Myths about Magnetic Media. DATAMATION No 3/1976, s. 65—70

[6] Gibbons T.: Integrity and Recovery in Computer Systems. NCC, Manchester 1976

[7] Guarding against the computer criminal. Computer International, ICL, April-June 1980, s. 13—14

[8] Hoffman L. J.: Modern Methods for Computer Security and Privacy. Prentice-Hall 1977

[9] Jaegermann T.: Metody technologicznej ochrony danych w statystyce państwowej. WIADOMOŚCI STATYSTYCZNE, nr 10/1980, s. 32—37

[10] Kotowski M.: Wstęp do ochrony danych. CINTE, Zeszyt 8/1979

[11] Marsh R.: Making Data More Secure. DATAMATION No 10/1976, s. 67—69

[12] Martin J.: Computer Data-Base Organization. Prentice-Hall 1977, s. 37—38

[13] Morris R., Thompson K.: Password Security: A Case History. Communications of the ACM, No 11/1979, s. 594—597

[14] Nagel K.: Obecny stan i nowości w dziedzinie bezpieczeństwa danych. Zeszyt 111 — Europejski Program Badawczy Diebolda, Warszawa 1980, s. 37—51

[15] Pantages A., Mc Lellan V., Myers E.: The Head-In-The-Sand Caper. DATAMATION No 9/1979, s. 70—82

[16] Patrick R.: 1. Sixty Ingredients for Better Systems. DATAMATION No 12/1977, s. 171—189

[17] Pinkerton J.M.M.: Security and Privacy of Data held in Computers. ICL Technical Journal, May 1980, s. 3—12

[18] Pritchard J.A.T.: Security in On-Line Systems. NCC, Manchester 1979

[19] Programm highlights (National Computer Conference). DATAMATION No 5/1979, s. 119—127

[20] Taber K. J.: On Computer Crime — Senate Bill S. 240. DATA PROCESSING DIGEST, No 1/1980, s. 1—2

[21] The security of managers information. Cunning Publications Inc. „EDP ALALYZER", No 7/1979, s. 1—13

[22] Weiss H.: Computer Security — an Overview. DATAMATION No. 1/1974, s. 42—47

[23] Yasaki E. K.: Security is a warm package. DATAMATION No 12/1978, s. 62.

Komputery w dydaktyce

Znaczenie zdobywania wiedzy jest ogromne. Wiedzę można przekazywać, ważniejsze jest jednak nauczenie zdobywania wiedzy. Znanych jest już wiele form nauczania, od dawna czyniono też próby opracowania specjalnych metod ułatwiających zdobywanie wiedzy.

W latach dwudziestych bieżącego stulecia Amerykanin S. Pressy zaprojektował specjalne maszyny do automatycznego nauczania. Jego idee nie znalazły jednak szerszego uznania. Po trzydziestoletniej przerwie podobne próby podjął B. F. Skinner, który — zamiast konstruowania specjalnych urządzeń do nauczania — zaproponował opracowanie metod sterowania procesem przyswajania wiedzy. Jego metody polegały głównie na przekazywaniu wiedzy w postaci odrębnych fragmentów, oddzielanych od siebie pytaniami, na które uczący się musi odpowiedzieć. Całość tak przygotowanego materiału nazywa się programem, a samo nauczanie — nauczaniem programowym.

Metody te uzasadniane były wynikami badań nad tresurą gołębi; zniechęcało to jednak tylko nielicznych. Zaczęto je więc stosować coraz powszechniej. Gdy zaś pojawiły się maszyny cyfrowe, przekształcono nauczanie programowane w nauczanie wspomagane komputerem. Przyczyną coraz większej popularności tego typu metod było powszechne przekonanie, że jest to jedyna metoda indywidualizacji nauczania. Nie bez wpływu pozostał także znaczny wzrost produkcji maszyn cyfrowych.

Charakterystyczną cechą rozwoju cybernetycznych metod nauczania w USA było dążenie do przekazywania jak największych „porcji" wiedzy w możliwie najkrótszym czasie i do tego — w sposób zindywidualizowany. W Polsce natomiast — jak to dobitnie wynika z materiałów konferencji INFOGRYF'80 w Kołobrzegu — głównym celem jest dążenie nie do dostarczania wiedzy, lecz automatycznego (komputerowego) jej kontrolowania, nie indywidualizacja

Tekst niniejszy jest skrótem większej wypowiedzi Autora (przyp. red.)

nauczania, lecz — unifikacja i algorytmizacja. Postuluje się więc przede wszystkim racjonalne (?), optymalne algorytmy nauczania. Szczególne zaś znaczenie przywiązuje się do kontrolowania i testowania. Proponowane są specjalne maszyny do testowania i programy generujące odpowiednio trudne zadania. Celem tych metod nie jest jednak sprawdzenie poziomu wiedzy lub jej jakości, lecz zmuszenie do jej zdobywania.

Metody przymusu w nauczaniu stosowano już w starożytnym Sumerze (rózgi). Cztery tysiące lat później w Anglii wykorzystywano specjalne pasy do chłosty, tzw. taws. Czasem była to zwykła linijka. Obecnie przy nauczaniu wspomaganym komputerowo mamy natomiast tzw. bloki testowo-informacyjne. Przykładowo:

„uzupełnij zdania: jednostka centralna wykonuje dwa podstawowe zadania: przetwarza wprowadzone in... według reguł arytmetycznych i ...gicznych, steruje pracą urządzeń ...trznych komputera. Zadanie to realizowane za pomocą bloku st..., bloku arytm..., bloku... operacyjnej.

Czas odpowiedzi: 30 s".

Mamy też specjalne „zobiektywizowane metody kontroli", przy czym „istotnym elementem metody jest model studenta. Model ten wiąże w funkcyjną zależność wielkości decydujące o doborze zadania o odpowiednim stopniu trudności" (z materiałów INFOGRYF-u'80).

Wszystko to robione jest z ogromną troską o dydaktykę. Wydaje się jednak, że nie nauczycielowi, lecz uczniowi należy pomagać, pamiętając o celu kształcenia — wyzwoleniu inicjatywy twórczej.

Po sprecyzowaniu celu nauczania trzeba wybrać odpowiednie środki. Może to być i maszyna cyfrowa. Ważne jest jednak samo nauczanie, a nie użyty do tego środek, podobnie jak w systemach nauczania „wspomaganych kredą".

Komputer także w jednych przypadkach bywa niezwykle przydatny w procesie nauczania, w innych — wręcz szkodliwy. Komputer może być użyty jako środek dodatkowy, uatrakcyjniający naukę i wpływający na lepsze przyswojenie materiału. Nie może jednak zastępować nauczyciela, którego nic nie zastąpi.

Jednym z ważniejszych środków wychowania jest osobowość nauczyciela, jego postawa, zachowanie, sposób bycia. Propaguje się dzisiaj indywidualizację procesu przyswajania wiedzy; z podobnych powodów należałoby dążyć do indywidualizacji form nauczania i wychowywania. Komputer może w tym pomóc, jeśli zostanie świadomie użyty.

Efektywne wykorzystanie komputera w procesie dydaktycznym wymaga odpowiedniej metodologii. Wykorzystanie maszyn związane jest z ich programowaniem, zatem metodologia ta zawierać powinna zarówno metodologię programowania, jak i metodologię nauczania, czyli — dydaktykę ogólną oraz dydaktyki szczegółowe. Jedną z naczelnych zasad tej metodologii powinno być — jak się wydaje — użycie komputera do nauczania, a nie testowania i kontrolowania. Nie oznacza to zanegowania metody zadań kontrolnych, nie można ich jednak formułować w postaci pytań-żądań z odpowiedzią na czas. Uczący się powinien mieć możliwość sterowania przebiegiem „lekcji komputerowej", tzn. realizacją programu. Analogicznie jak przy lekturze książki, kiedy czytający sam decyduje o tym w jakiej kolejności czytać, na co zwrócić szczególną uwagę, a które fragmenty opuszczać.

Oczywiście, nauczyciel może (i powinien) dawać rady i wskazówki, które jego zdaniem ułatwią rozumienie i przyswojenie materiału. Nie może on jednak rościć sobie prawa do apodyktycznego (poprzez narzucenie programu) sterowania procesem przyswajania informacji, trzeba bowiem zakładać ewentualną wyższą inteligencję nauczanego, aniżeli uczącego i sterującego.

W procesie nauczania pytania powinien stawiać przede wszystkim uczący się. Tu właśnie komputer może się okazać bardzo przydatny. Nie wstydząc się swej niewiedzy i nie bojąc „rózg" nauczyciela, uczeń może pytać o to, co powinien przeczytać, co musi umieć, jakie jest znaczenie nieznanego mu słowa itp. Tu dopiero nauczyciel może wykazać swoje umiejętności, inwencję, wiedzę, a nie swoją przewagę instytucjonalną. A zatem nauczyciel powinien być też dobrym programistą. To z kolei implikuje konieczność stosowania prawdziwych języków programowania do zapisu lekcji (do ich programowania), nie zaś opracowywanych ad hoc przeróżnych notacji zwanych j ę z y k a m i a u t o r s k i m i. Jaki jest ten „prawdziwy" język programowania na razie nie bardzo wiadomo. Niewątpliwie musi to być uniwersalny język (nie specjalistyczny), nieskomplikowany, łatwy w nauce, umożliwiający układanie programów (tzn. zapisywanie jednostek dydaktycznych) w sposób systematyczny, zrozumiały i czytelny (strukturalnie).

Postulat języka uniwersalnego przesądza oczywiście o uniwersalności opracowywanych systemów nauczania. Tak więc absolutnym nieporozumieniem jest tworzenie systemów nauczania języka FORTRAN, systemów nauczania algebry liniowej itp. Mogą one co najwyżej stanowić pewną liczbę programów (niech będą nawet nazwane pakietami czy podsystemami), wykonywanych w ramach jednego systemu zarządzającego ich przechowywaniem, administracją, wywoływaniem, wykonywaniem (oczywiście w trybie dialogu) itd. Do realizacji takiego systemu najlepszy byłby mały komputer do wyłącznej dyspozycji systemu nauczania lub tzw. komputery osobiste, podłączone do komputera ogólnego.

Walenty OSTASIEWICZ

KONFERENCJE

CAMAC w zastosowaniach przemysłowych

W dniach 8—9 października br. w Warszawie (budynek NOT, ul. Czackiego 3/5, sala A) zostanie zorganizowana konferencja, związana z aparaturą systemu CAMAC w zastosowaniach przemysłowych. Organizatorem konferencji jest Komitet ds. Systemu CAMAC przy ZG SEP. Na konferencji zostaną wygłoszone krajowe i zagraniczne referaty na temat tendencji rozwojowych w modularnych systemach sterowania i pomiarów, ze szczególnym uwzględnieniem systemu CAMAC; omówione zostaną doświadczenia wynikające z dotychczasowych zastosowań CAMAC w systemowych procesach automatyzacji oraz problemy sterowania w systemie CAMAC.

Konferencja posłuży wymianie doświadczeń pomiędzy specjalistami zajmującymi się rozwojem, produkcją i zastosowaniem systemu CAMAC a użytkownikami tego systemu.

Informację nt. warunków uczestnictwa można uzyskać u **mgr. inż. Jana Jacha, tel. 27-24-12, telex 812477, Biuro Zbytu Zjednoczonych Zakładów Urządzeń Jądrowych POLON, ul. Bielańska 1, 00-086 Warszawa.**

O komputerach w automatyce

Spośród 20 sesji tematycznych VIII Krajowej Konferencji Automatyki (Szczecin, 16—17 września ub.r.) tylko kilka dotyczyło roli komputerów w automatyce. Ponieważ trudno już sobie dzisiaj wyobrazić automatyzację bez użycia komputera, kalkulatora, mikroprocesora lub choćby układu cyfrowego, warto poświęcić nieco uwagi przedstawionym referatom.

ELEMENTY I UKŁADY

Elementy i układy automatyki cyfrowej przedstawiono na sesji 13. W moim przekonaniu stanowiła ona odbicie stanu aktualnego oraz tendencji rozwoju sprzętu komputerowego. W referatach, które dotyczyły bezpośrednio zagadnień praktycznych, uwzględniono bardzo różnorodne rozwiązania, podając w większości opisy konkretnych realizacji.

Podkreślano nieuchronność zastosowania w automatyce układów o dużym stopniu scalenia (a przede wszystkim — mikroprocesorów) do budowy regulatorów i systemów sterowania (W. Koziński, R. Swiniarski, Instytut Sterowania i Elektroniki Przemysłowej PW). Zwracano uwagę na wzrost znaczenia normalizacji układów sprzęgających komputer z obiektem, np. INTELDIGIT-PI lub CAMAC (W. Kozłowski, Przemysłowy Instytut Automatyki i Pomiarów; J. Kierzek i in., Instytut Badań Jądrowych, Warszawa). Sprawa normalizacji sprzętu (elementów i układów), choć istotna, jest jednak jeszcze niedostatecznie uświadamiana.

Autorzy opisywali na ogół układy zrealizowane za granicą lub — w kraju — na importowanym czy licencyjnym sprzęcie. Układy produkcji krajowej prezentowano także; ponieważ jednak „warunek oparcia konstrukcji o aktualnie produkowane w kraju elementy (założenie samo w sobie bardzo chwalebne, przyp. JZ) wykluczył możliwość zastosowania systemu mikroprocesorowego", zastosowanie elementów o mniejszym stopniu scalenia może nas cofnąć do poprzedniej epoki, choćby nawet w prostych układach ich użycie było całkowicie uzasadnione. Prowadzi to do podstawowego i oczywistego wniosku, że brak mikroprocesorów rodzimej produkcji jest barierą uniemożliwiającą prawidłowy rozwój elementów i układów automatyki. Na różnych sesjach Konferencji podkreślano fundamentalne znaczenie niezawodności sprzętu, co nadal nie dociera do producentów. W tej sytuacji należy stwierdzić, że możliwości wyboru elementów i układów dla automatyki cyfrowej są więcej niż skromne. Czy na istniejącym sprzęcie można coś zrobić i jakim kosztem? Odpowiedź na to pytanie dał przebieg sesji 11, dotyczącej zastosowań komputerów.

ZASTOSOWANIE KOMPUTERÓW DO STEROWANIA

O możliwościach sprzętu komputerowego świadczą prace J. Sołtysika i in. (Instytut Informatyki PG). W trzech referatach autorzy ci przedstawili bogate doświadczenie zdobyte podczas budowy komputerowych systemów automatyzacji pomiarów i sterowania. Ponieważ wykonali i opisali trzy systemy — od zakupu sprzętu (zwykle przez użytkownika), przez wytworzenie oprogramowania i jego dokumentacji, aż do fazy uruchomienia i przekazania systemu w środowisku docelowym — warto przekazać Czytelnikom więcej informacji na ten temat.

Z rozważań autorów wynika, że w fazie zakupu sprzętu należy wziąć pod uwagę nie tylko właściwości funkcjonalne, jakie musi mieć system komputerowy spełniający zadanie automatyzacji. Równie ważne — lecz prawie nigdy nie brane pod uwagę przez użytkowników — jest znaczenie zasad programowania oraz modyfikacji i niezawodności sprzętu. Wskutek niedostatecznego uwzględnienia tych czynników cierpią programiści (i projektanci) systemu, a przez brak niezawodności — także sami użytkownicy.

Pionierskie — w warunkach polskich — jest wyraźne wyodrębnienie fazy specyfikacji układu sterowania, której autorzy poświęcili odrębny referat (nazywając ją algorytmizacją). Brak przekonania użytkowników o potrzebie wyraźnej formalizacji wymagań prowadzi do niekompletności, nieczytelności i wewnętrznej sprzeczności ich opisu. W konsekwencji formułowanie wymagań może się przeciągnąć aż do fazy uruchamiania zestawu.

Istotne jest odróżnienie zestawu narzędzi, służących do budowy systemu, od jego konfiguracji docelowej oraz — bieżące prowadzenie dokumentacji. Niedoskonałość narzędzi (mówiąc dobitniej ich całkowity brak), podobnie jak niepełna specyfikacja i zawodność sprzętu są najczęstszymi przyczynami błędów, wpływając na jakość produktu końcowego.

Jeżeli testowanie i uruchamianie zestawu przebiega całkowicie w warunkach przyszłego użytkowania, a użytkownicy zapewnili dostateczną wielkość nakładów na przekazanie i utrzymanie systemu, to można mieć nadzieję, że system spełni ich oczekiwania. Szczególnie w końcowej fazie realizacji projektu, ogromne znaczenie ma szybka komunikacja między użytkownikami i autorami systemu, co — dodajmy — w krajach o najbardziej rozwiniętej technologii odbywa się najwydajniej za pomocą sieci teleinformatycznej.

Są to przykładowe problemy, z jakimi mogą się zetknąć realizatorzy systemów automatyki w jakiejkolwiek dziedzinie. Jeżeli ktoś ma zamiar automatyzować badania z użyciem komputera, to musi zapoznać się z wynikami prac autorów z Gdańska, które stanowiły jeden z jaśniejszych punktów VIII Krajowej Konferencji Automatyki.

Zagadnienia komputeryzacji badań omówiono jeszcze w kilku innych referatach (na innych sesjach), z których warto wymienić dwa: „Kompleksowy system automatyzacji eksperymentu" (H. Kordecki i in., Instytut Cybernetyki Technicznej PWr.) i „System pomiarowo-diagnostyczny elektronicznych zespołów funkcjonalnych ..." (R. Zielonko i in., Instytut Technologii Elektronowej PG). Ich treść świadczy o osiągnięciu przez autorów znacznego poziomu kompetencji i — o konsekwencji w rozwiązywaniu zagadnień automatyzacji.

W pierwszym z tych referatów przedstawiono system o nazwie EXPES, złożony z minikomputera wraz ze standardowymi urządzeniami zewnętrznymi i oprogramowaniem firmowym, poszerzony o wyspecjalizowane urządzenia wejścia — wyjścia i oprogramowanie użytkowe. Ze względu na zbliżony charakter badań w różnych dziedzinach nauki, tj. w fizyce, chemii,mechanice lub elektryce, system można uznać za uniwersalny. Dotychczas zrealizowano — lub są w trakcie realizacji — zestawy złożone z następujących minikomputerów i sprzęgów: ODRA 1325 — SMA, MERA 400-INTELGIDIT PI, SM 3 — CAMAC z bogatym oprogramowaniem użytkowym, głównie do identyfikacji obiektów. W przyszłości autorzy zamierzają zwiększyć możliwości zestawów przez rozszerzenie lub zmianę systemów operacyjnych, a także przez połączenie minikomputerów w sieć np. z wykorzystaniem komputera R-32 jako składnika centralnego ze wspólną bazą danych.

Drugi z wymienionych referatów dotyczył realizacji systemu pomiarowego nazwanego DELTA 022, zbudowanego w oparciu o minikomputer MERA 400 bez dysku, z systemem operacyjnym CROOK i językiem BASIC. Autorzy dysponują również niemałym doświadczeniem w realizacji kompute-

rowych systemów pomiarowych[1]) i mogliby podjąć próbę wyjaśnienia innym, jak można w tak trudnych warunkach i przy tak wielu ograniczeniach zrealizować system komputerowy spełniający wymagania użytkowe.

Fakt, że we wszystkich wymienionych systemach stosowano prawie wyłącznie sprzęt polski, nie może jednak prowadzić do wniosku, że produkujemy aż tak dobre komputery. Odpowiedź na to pytanie, ile są one warte, nie jest jednoznaczna. Jeżeli w pierwszym z wymienionych systemów „nie wykorzystano żadnych elementów oprogramowania firmowego", uznałbym to za tragedię dla firmy. Aby uzyskać pełniejszą odpowiedź, podajmy (wg referatu M. Loręckiego, Instytut Gospodarki Magazynowej i T. Puchałki, Instytut Automatyki PP), jakie cechy powinien mieć minikomputer do sterowania (choć w odniesieniu do jednego tylko zastosowania). Są to:

- elastyczność w doborze struktury, zapewniona przez wspólną szynę
- bogaty zestaw oprogramowania podstawowego, w tym standardowy, dyskowy system operacyjny czasu rzeczywistego
- bogaty zestaw urządzeń peryferyjnych, umożliwiający łatwe tworzenie stanowisk operatorskich
- wysoka jakość wykonania i niezawodność działania
- dobrze zorganizowany serwis, zapewniony przez producenta.

Konia z rzędem temu, kto udowodni, że którykolwiek z tych warunków jest spełniony przez komputery krajowe. Chylę czoło przed ludźmi, którzy potrafili na tym sprzęcie tyle zrobić. Sukces swój zawdzięczają nie tyle jego jakości, ile własnej wytrwałości i zaangażowaniu niewspółmiernie dużych środków.

Zagadnienia komputeryzacji badań wysunąłem tu na plan pierwszy, ponieważ wydaje mi się, że umknęły one uwadze organizatorów konferencji. Tymczasem problematyka dojrzała już do tego, aby poświęcić jej odrębną sesję i to zestawioną nie przypadkowo (jak niektóre inne sesje tej konferencji), lecz — z rozmysłem. Oczywiście, w materiałach konferencyjnych można się doszukać interesujących zastosowań komputerów do sterowania w przemyśle, jak choćby w hutnictwie (W. Hejmo i in., Huta im Lenina), w energetyce (J. Dziedzic, W. Kosmowski, Instytut Elektroenergetyki i Automatyki PG) i w innych dziedzinach gospodarki. Charakterystyczne jest, że w tych zastosowaniach wykorzystuje się głównie komputery importowane, a tylko wyjątkowo — krajowe. Jest to zrozumiałe, jeżeli się zważy, że w przemyśle komputery są środkami produkcji, a te nie mają racji bytu, gdy się ciągle psują.

[1]) Hoja A., Szczypta A., Tlaga W., Zielonko R., INFORMATYKA nr 1/79; Zielonko W., ELEKTRONIKA nr 11/80.

OPROGRAMOWANIE PODSTAWOWE

Spośród prac związanych z zastosowaniem można wyróżnić sporą grupę referatów dotyczących oprogramowania komputerów do sterowania.

Jeden z referatów plenarnych dotyczył badań prowadzonych w zakresie komputerowych systemów sterowania (H. Górecki, A. Gościński, Instytut Informatyki i Automatyki AGH). Droga do celu ostatecznego, tj. do podjęcia badań nad zastosowaniem komputerowych układów automatyki w przemyśle, wiodła przez dwa etapy pośrednie, a więc: wykształcenie studentów i pracowników pod kątem zastosowania metod automatyki przy użyciu komputerów (czego efektem było opublikowanie książki[2]) oraz zbadanie możliwości wykorzystania komputerów do rozwiązywania problemów naukowych. Niektóre wyniki szczegółowe obecnie prowadzonych prac zostały przedstawione w dwóch innych referatach (A. Gościński i in.), dotyczących rozszerzania języka modelowania zdarzeń w celu sterowania procesami w czasie rzeczywistym oraz określenia momentów aktywizacji programów sterowania bezpośredniego i optymalizacji przy wykorzystaniu systemu operacyjnego komputera sterującego.

Ważne problemy oprogramowania podstawowego podejmowali — na sesji dotyczącej zastosowań — autorzy z ośrodka wrocławskiego. Zarządzanie pamięcią operacyjną, rozdział zasobów czy optymalne szeregowanie zadań (H. Kordecki, W. Myszka, A. Stanisz, Instytut Cybernetyki Technicznej PWr.) w komputerowych systemach sterowania, to zagadnienia nie rozwiązane dotąd w sposób dostatecznie ogólny. Ich podjęcie świadczy niewątpliwie o właściwym rozpoznaniu problematyki i należy mieć nadzieję, że autorzy na tym nie poprzestaną.

Jakkolwiek trudno przecenić rolę i potrzebę dobrego oprogramowania podstawowego, to nie ma na nie odpowiedniego zapotrzebowania, a więc nikt w nie nie inwestuje i powstaje wrażenie braku koordynacji. Pojedyncze prace nie zmienią obrazu sytuacji. Metod badawczych dostarczyły prace nad językami programowania, których stworzenie wymaga — jak wiadomo — pracy silnych zespołów realizacyjnych.

Konieczna jest zatem koordynacja prowadzonych prac teoretycznych nad oprogramowaniem podstawowym oraz skuteczne ich kończenie, powstaniem systemów operacyjnych lub translatorów, które mogą sprostać stawianym warunkom. Jednak zainteresowani tym powinni być przede wszystkim producenci. Pilne jest także rozpoczęcie skoordynowanych prac nad językiem programowania systemów czasu rzeczywistego. Wcale nie musi to być od razu ADA, wystarczyłby PEARL, FORTH lub choćby IRT-FORTRAN lub RT--BASIC, byle by koncepcja nie była wzięta z sufitu. Niemcy i Amerykanie wiedzieli jak to robić już dziesięć lat temu i robią nadal — nie szczędząc pieniędzy i wysiłku.

[2]) H. Górecki i in.: Algorytmy i programy sterowania. WNT, Warszawa, 1980.

OPROGRAMOWANIE UŻYTKOWE

W tej dziedzinie sytuacja jest lepsza, ponieważ stworzenie systemu programów przeznaczonych do pojedynczego zadania jest mniej pracochłonne i nie wymaga zbyt dużych nakładów.

Na konferencji przedstawiono niejeden taki system programów, jak choćby uniwersalny pakiet programów optymalizacji statycznej, napisany w języku BASIC na kalkulator HP 9830A by (T. Kuźnik, Instytut Metalurgii Żelaza, Gliwice), czy bibliotekę programów obliczania transmitancji widmowych typowych obiektów chemicznych (M. Metzger, Instytut Automatyki PSl), przeznaczoną do pomocy w rozpoznawaniu właściwości regulacyjnych tych obiektów w układach sterowania bezpośredniego. Omówiono także (J. Kolendowski, J. Budziaszek, Cyfronet, Kraków) dostępne na maszynie cyfrowej CYBER 72 trzy języki (programy): CSSL, MIMIC i FORSIM, umożliwiające symulację prawie wszystkich zagadnień z zakresu modelowania układów ciągłych, jakie mogą pojawić się w Polsce. Mają one tę wadę, że można z nich korzystać tylko w jednym, najwyżej w dwóch ośrodkach w kraju. Tym bardziej dziwne jest, że nie ma dotąd pakietu, który działałby na polskim komputerze i objął zasięgiem cały kraj. Obrazuje to — jak mi się wydaje — stan i organizację współpracy ośrodków krajowych.

Niezwykle ważnym zagadnieniem automatycznego projektowania poświęcono tylko jeden referat dotyczący projektowania systemów pomiarowych (J. Nalepa, M. Szyper, Instytut Maszyn i Sterowania Układów Elektroenergetycznych AGH). Nie znalazłem natomiast ani jednego referatu, który dotyczyłby komputerowego projektowania układów regulacji, choć sam problem jest niezwykle istotny.

Wiele przedstawionych zagadnień obliczeniowych, w tym także metody symulacji, nie stanowi o specyfice zastosowania komputerów w automatyce, zatem nie będę ich omawiał, podobnie jak — znacznej liczby algorytmów przedstawionych na konferencji.

UKŁADY ROZŁOŻONE PRZESTRZENNIE

Zagadnienia układów rozłożonych przestrzennie omawiano na sesji 9, zatytułowanej „Sieci teleinformatyczne i telemechanika". Jeżeli dobrze rozumiem na czym polega rola systemów informatycznych w automatyce, to obrady tej sesji dotyczyły tylko w części tych zagadnień.

Interesująco i z dużą znajomością rzeczy omówił niektóre problemy R. Jezierski (Instytut Automatyki PP), przedstawiając w dwóch referatach realizację współpracy jednostek centralnych systemu MERA 300 i uniwersalny moduł sprzęgający urządzenia zewnętrzne z kanałem multipleksera tego systemu. Szkoda tylko, że jest to koncepcja jednorazowa i cały układ należy zaprojektować na nowo, gdy zmienią się jednostki wymieniające informację.

Godne uwagi prace prowadzone w tym samym ośrodku nad systemami TM-11 i TM-10 (S. Grocholewski i in.) przedstawiono w czterech innych referatach. Dziwi mnie, że „zrezygnowano z zastosowania jakiejkolwiek ze znanych magistral mikrokomputerowych, na rzecz wprowadzenia sygnałów odciążających w dużym stopniu udział oprogramowania i pracę projektantów". Czy nie dlatego właśnie wprowadzono magistrale standardowe?

Wydaje się, że trwające od kilku lat prace organizacji międzynarodowych nad normalizacją komputerowych układów sterowania rozłożonych przestrzennie są warte tego, aby zwrócili na nie uwagę autorzy krajowych opracowań. Zgodność z normami światowymi jest tu nieodzowna, aby można było stosować urządzenia produkcji różnych firm (co przy ciągłym niedostatku produkcji krajowej ma znaczenie niebagatelne) i może przynieść tylko korzyści. Dowodów na to dostarcza choćby system CAMAC.

* * *

Referaty plenarne i przeglądowe Konferencji zostaną dopiero opublikowane, nie powinny one jednak w sposób zasadniczy zmienić przedstawionego tu obrazu, choć z pewnością go uzupełnią i rozszerzą.

Postawienie pełnej diagnozy i wskazanie rozwiązań jest niezwykle trudne. Jednak dysponując bogatym materiałem Konferencji, należy wskazać niektóre przyczyny słabości w stosowaniu komputerów do sterowania. Inaczej praca projektantów i konstruktorów będzie nadal marnotrawiona, bo co też począć z projektami, których nie można zrealizować przy użyciu krajowego sprzętu lub krajowych elementów.

Należałoby przede wszystkim odpowiedzieć na pytanie, na czym i czym ci ludzie mają pracować? Z pewnością odpowiedź będzie łatwiejsza, gdy nadrobimy zaległości technologiczne w produkcji elementów oraz rozpoczniemy skoordynowane prace nad oprogramowaniem podstawowym sprzętu, który trzeba tak dopracować, aby osiągnął wystarczającą niezawodność.

Charakterystyczne, że ta odpowiedź nie odnosi się tylko do automatyki i mogłaby być jeszcze krótsza, gdyby ktoś zdecydował się odnieść ją do miejsca wyznaczającego aktualny stan potencjału komputerowego kraju. Jej istotną wadą jest natomiast to, że nie uwzględnia istotnych czynników organizacyjno-ekonomicznych rozwoju tego potencjału, lecz ten aspekt niestety nie podlega kontroli informatyków czy automatyków.

Janusz ZALEWSKI

Jaki powinien być nowy polski minikomputer?

Na to pytanie starali się odpowiedzieć uczestnicy czterogodzinnego seminarium zorganizowanego 24 kwietnia br. w Instytucie Maszyn Matematycznych MERA-IMM w Warszawie. Grupa specjalistów z IMM, Instytutu Informatyki UW oraz Zakładów MERA-System (pracująca pod kierunkiem mgr E. Jezierskiej-Ziemkiewicz) przedstawiła i poddała pod dyskusję wstępną koncepcję systemu minikomputerowego, nazwanego roboczo SOLID.

Celem seminarium przygotowanego już w początkowym etapie prac nad systemem, było:

- poinformowanie środowiska o wstępnych założeniach SOLID
- zainteresowanie współpracą innych ośrodków (konieczną dla pełnej realizacji projektu).

Należy wspomnieć, że projekt dotyczy minikomputera możliwego do wprowadzenia za kilka lat, powiedzmy — do roku 1985.

W pierwszych dwóch referatach przedstawiono wymagania stawiane architekturze systemu. Wymagania eksploatacyjne sprzętu i oprogramowania obejmują:

- przystosowanie do zestawiania systemów przez użytkownika, tworzenia różnorodnych zestawów — od struktur najprostszych, przez sieci lokalne, do złożonych systemów cyfrowych o dużej mocy obliczeniowej
- umożliwienie szybkiego wytworzenia oprogramowania i późniejszej łatwej pielęgnacji (ang. **maintenance**) a także — możliwości przenoszenia całego oprogramowania (tzn. zarówno programów użytkowych, jak i systemów operacyjnych)
- osiągnięcie dostatecznej niezawodności sprzętu i oprogramowania oraz prostoty obsługi.

Z przyczyn ekonomicznych wynikają następne wymagania: zapewnienie użyteczności częściowych wyników realizacji projektu na każdym etapie prac i uniknięcie ograniczeń uniemożliwiających przystosowanie do nieprzewidzianych zmian w technologii elementów lub w organizacji komputerów. Można to sprowadzić do stwierdzenia, że na każdym etapie realizacji projekt powinien być otwarty, dający się przystosować do aktualnych tendencji światowych czy krajowych możliwości.

Okazuje się, że tak ogólne wymagania i przewidywane różnorodne możliwości zastosowań (np. do opracowywania oprogramowania, w systemach graficznych, w sytemach telekomunikacyjnych, dydaktycznych, w bankach danych, do sterowania procesami itp.) narzucają dość konkretne warunki z punktu widzenia zasad konstrukcyjnych. Zostawiając ich szczegółowe wyjaśnienie autorom projektu, w niniejszym artykule zasygnalizujemy jedynie dwie rzeczy.

Po pierwsze — konieczną elastyczność struktury systemu autorzy zamierzają zapewnić przez skonstruowanie tzw. multikomputera, o organizacji mającej cechy systemu wieloprocesorowego i wielokomputerowego (o dokładniejsze sformułowanie należy poprosić autorów, ponieważ dla mnie i dla niektórych uczestników seminarium znaczenie słowa m u l t i k o m p u t e r nie jest jasne).

Po drugie — całość oprogramowania musi powstać w języku strukturalnym wysokiego poziomu, co skłania także do ukierunkowania architektury systemu na realizację układową tego języka.

Nie można wiele powiedzieć o realizacji układowej minikomputera, ponieważ trudno w tej chwili rozstrzygnąć wszystkie szczegóły architektury i organizacji, choćby ze względu na niewiadome technologiczne. Przeciwnie jest w przypadku oprogramowania, do realizacji którego wybrano język C. Dlatego aż sześć kolejnych referatów poświęcono omówieniu różnych aspektów oprogramowania systemu, począwszy od samej koncepcji i zarysu systemu operacyjnego, przez właściwości systemu plików i systemu produkcji translatorów, aż do samego języka C i C-kodu. Język ten — choć bliżej w Polsce nie znany — jest już sprawdzony na wielu różnych maszynach i posłużył m.in. do napisania systemu operacyjnego UNIX (Bell Laboratories, Murray Hill, NJ, USA — dla komputerów PDP-11), który pod względem funkcjonalności przewyższa oryginalne systemy operacyjne firmy DEC.

Sprawy najistotniejsze dla całego przedsięwzięcia, tj. dotyczące jego realizacji, omówiono w trzech końcowych referatach. Ponieważ istniejące oprogramowanie dla minikomputerów produkcji krajowej lub wykorzystywanych w kraju jest dalekie od doskonałości, autorzy proponują w wariancie minimalnym opracowanie kompletnego oprogramowania opartego na języku C, np. dla MERY 400, z jednoczesnym wprowadzeniem niezbędnych zmian konstrukcyjnych.

W pierwszym etapie konieczne będzie także zbudowanie narzędzi do wytwarzania oprogramowania, a szczególnie — systemów graficznych. Z chwilą opracowania systemu operacyjnego i translatora języka C dla jednego z typowych minikomputerów używanych w kraju, można rozpocząć prace nad generatorem translatorów. Przedsięwzięcie w pełnym wymiarze (a więc program maksimum) można zrealizować budując nowy minikomputer (multikomputer SOLID), do czego jednakże — przy świadomej rezygnacji z importu elementów — niezbędny jest rozwój prac nad polską bazą elementową, a głównie — nad układami o dużym stopniu scalenia, w tym — matrycami programowanymi.

Autorzy są świadomi, że nawet przy spełnieniu tych warunków do pełnego zadowolenia będzie daleko, jeżeli nie rozpocznie się równocześnie produkcji urządzeń peryferyjnych w szerokim zakresie oraz nie usprawni dystrybucji, serwisu, szkolenia, dokumentacji i działalności wydawniczej.

W dyskusji zabierali głos użytkownicy i konstruktorzy. Ci pierwsi zwracali uwagę przede wszystkim na potrzebę rozpoczęcia sprzedaży urządzeń zewnętrznych i rozszerzenia ich asortymentu, a także — znacznego zwiększenia niezawodności tych urządzeń i poprawę serwisu.

Natomiast głosy konstruktorów dotyczyły istotnych problemów technicznych o znaczeniu podstawowym, jak np. zapewnienia bazy elementowej i zbadania możliwości kooperacyjnych, kwestii języka C i generatorów oprogramowania. Bardziej szczegółowe rozważania postanowiono kontynuować na seminariach specjalistycznych w IMM. Z grona konstruktorów pochodzą także głosy o konieczności dokładnego rozpoznania potrzeb, a także efektów, jakie może przynieść komputeryzacja. Zaproponowano również szybką realizację sprawnego, prostego systemu minikomputerowego, który mógłby stanowić remedium na niektóre bieżące bolączki naszego życia. Warunkiem skuteczności wprowadzenia tego systemu musiałoby być szybkie jego skonstruowanie i prostota obsługi. Cel skonstruowania takiego systemu należy rozumieć jako chęć zapełnienia luki między skomplikowanym minikomputerem a możliwością znacznych usprawnień przy użyciu prostych środków obliczeniowych.

W podsumowaniu wyników seminarium trzeba powiedzieć, że prace nad nowym polskim minikomputerem należy zacząć od uporządkowania bałaganu w sferze sprzętu informatycznego, oprogramowania, serwisu i sprzedaży. Jeżeli niejednorodność oprogramowania, jego niedoskonałość, braki w dokumentacji i pielęgnacji są tak duże, jak to przedstawiono, to z całą pewnością należy prowadzić prace (skoordynowane!) nad systemem programowania opartym o język strukturalny. Czy ma to być język C, powinni rozstrzygnąć specjaliści, choć na seminarium nie słyszałem głosów sprzeciwu.

Do realizacji prac nad nowym komputerem praktycznie nie można przystąpić bez posiadania odpowiednich narzędzi, tj. w pierwszym rzędzie — systemów graficznych, wyposażonych w monitory ekranowe, pisaki itp. Bez takich systemów, jak również bez urządzeń testujących, czeka krajowych producentów układów elektronicznych o dużym stopniu scalenia poważnie utrudnione zadanie. Z kolei należy pamiętać, że odpowiedni poziom krajowej produkcji mikroprocesorów, pamięci półprzewodnikowych oraz matryc programowanych jest podstawowym warunkiem skuteczności prac nad projektem nowego systemu minikomputerowego, a więc — losu całego przedsięwzięcia.

Ponadto należy zwrócić uwagę, że jest to ogromne zadanie, które wymaga doskonałej koordynacji. Z inżynierskiego punktu widzenia realizacja tego projektu jest uzasadniona, natomiast istnieje obawa, czy uda się osiągnąć cel posługując się dotychczas stosowanymi metodami organizacji i ekonomiki. Zorganizowanie seminarium wydaje się pierwszym, choć małym krokiem do zmiany tych metod.

Janusz ZALEWSKI

ZWIĄZKI ZAWODOWE

Zreformowane przedsiębiorstwo informatyczne — podstawowe kierunki przemian

Przygotowanie i wdrożenie kompleksowej reformy w całej gospodarce narodowej wymaga zapewne sporo czasu. Wymaga też licznych wstępnych przedsięwzięć, minimalizujących ryzyko doszczętnej dewastacji więzi kooperacyjnych i handlowych oraz niekontrolowanego wybuchu inflacji i bezrobocia. Wydaje się jednak, że istnieją obszary działalności gospodarczej, w których po pierwsze — reforma może być wprowadzana niemal „od razu", a po drugie — może być ona znacznie bardziej radykalna niż w przypadku innych dziedzin. Takim obszarem jest — między innymi — informatyka. Podstawowe przesłanki skłaniające do tego wniosku są następujące.

● Działalność informatyczna jest — w sensie „technologicznym" — niemal zupełnie niezależna od zewnętrznych procesów produkcyjnych i wytwórczych, tzn. od najszerzej rozumianej „wydolności" otoczenia gospodarczego.

● Autonomia informatyki jest również wyraźna w sferze czysto ekonomicz-

nej: koszty usług informatycznych są w minimalnym stopniu kosztami ciągnionymi i powstają w przeważającej mierze w samym przedsiębiorstwie informatycznym; z kolei ceny tych usług płacone przez użytkowników nie stanowią w zasadzie żadnego ogniwa większego łańcucha kosztowo-cenowego.

● Metody zarządzania przyjęte w przedsiębiorstwach informatycznych nie są uwarunkowane obowiązującymi rozwiązaniami poza informatyką, ani też nie wymuszają na otoczeniu żadnych określonych zasad organizacji i zarządzania.

● Obserwowany spadek zainteresowania informatyką ze strony organizacji gospodarczych od chwili ich „lekkiego" usamodzielnienia — świadczy, że „reanimowanie" informatyki metodami administracyjnymi jest działalnością nieskuteczną, szkodliwą, przynoszącą jedynie straty gospodarcze.

● Poziom cen usług informatycznych, ukształtowany obecnie w wyniku absurdalnych cen sprzętu, jest ewidentnie zawyżony w stosunku do kosztów niezbędnych dla wytworzenia tych usług.

Jeżeli powyższe uwagi są prawdziwe, niekwestionowana musi być również słuszność opisanych niżej implikacji.

● Autonomiczność usług informatycznych jest tak wielka, że ryzyko powstania niekontrolowanych perturbacji, związane z najbardziej nawet rewolucyjną reformą w informatyce, ogranicza się do samej informatyki.

● Informatykę można potraktować jako swoisty „balon próbny" reformy, tzn. reformować ją, nie czekając na zreformowanie innych dziedzin gospodarki. Poczynione obserwacje pozwoliłyby poznać niemal pełną gamę społecznych i gospodarczych skutków tych lub innych rozwiązań.

● Zastosowania informatyki w gospodarce krajów rozwiniętych, a także doświadczenia rodzime dają jednoznaczny dowód na to, że ani potrzeb na usługi informatyczne, ani roli tych usług nie można ustanowić mocą dekretu lub sugestii. Informatyka polska musi się dzisiaj sama sprawdzić i wykazać swoją przydatność. Administracyjne metody pobudzania zainteresowania informatyką kompletnie zawiodły. Konieczne jest więc wypróbowanie innych sposobów organizacji świadczenia i sprzedaży usług informatycznych.

● Niedostateczne wykorzystanie sprzętu oraz zmniejszanie się popytu na usługi informatyczne upewnia, że wprowadzenie rynkowych zasad gospodarowania usługami informatycznymi spowodowałoby ukształtowanie rynku klienta, użytkownika (a nie producenta), co byłoby w skali całej gospodarki znaczącym korzystnym ewenementem. Nie miałyby tu zatem racji bytu podnoszone w związku z przewidywaną reformą obawy wzmocnienia struktur monopolistycznych. W przypadku informatyki radykalna reforma może przynieść wyłącznie skutek odwrotny.

● Rewolucja cenowa w informatyce, posiłkująca reformę struktur zarządzania, wzmocniłaby efekty opisane powyżej. Jedynym mającym racjonalne uzasadnienie kierunkiem zmian cen jest ich znacząca obniżka. Oznacza to, że informatyka — znów jako jeden z nielicznych wyjątków — nie miałaby swego udziału we wzmaganiu presji inflacyjnej. Ponadto można by dzięki temu uzyskać wzrost popytu na informatykę, możliwy do zaspokojenia i zrównoważenia poprzez pełniejsze wykorzystanie już istniejących zasobów, bez wtórnego wzrostu cen. A tylko dzięki wzrostowi autentycznego popytu da się zminimalizować redukcje kadr informatycznych, a także wielkość — już w tej chwili przemożnego — ukrytego bezrobocia.

● Informatyka jest jednym z elementów infrastruktury gospodarczej. A to, czy — jako składnik tej infrastruktury — stanie się moderatorem gospodarki (czynnikiem przyspieszającym i regulującym jej rozwój), czy też nadal pozostanie kosztownym ozdobnikiem — zależy od stopnia przystosowania metod informatyki i kierunków jej ekspansji do autentycznych potrzeb gospodarki. Artykulacja tych potrzeb „wprost" jest — jak wykazała praktyka — niemożliwa; jedyną skuteczną wydaje się metoda „nadążna", przeprowadzana na zasadzie prób i błędów — ciągłe dostrajanie się usług informatycznych do zgłaszanego zapotrzebowania. Aby informatyka — w chwili powszechnego wprowadzania reformy — mogła stać się naturalnym komponentem gospodarki i użytecznym jej narzędziem, powyższa wytyczna musi być już wówczas ugruntowaną zasadą usług informatycznych. Jedyną szansą osiągnięcia tego stanu jest wcześniejsze niż w całej gospodarce — wdrożenie reformy w informatyce. W przeciwnym przypadku, gdy gospodarka „ruszy" bez informatyki, zakwestionowanie przez praktyków gospodarczych jej użyteczności nie powinno być dla nikogo niespodzianką.

Analiza aktualnego stanu zastosowań informatyki w Polsce (pogłębiający się regres i brak jakichkolwiek perspektyw poprawy w ramach obecnych struktur organizacyjno-ekonomicznych) oraz bardzo nieznaczne, ograniczone ryzyko ewentualnego niepowodzenia reformy informatyki przy jednoczesnych niebagatelnych profitach: wzbogaceniu asortymentu usług informatycznych, uelastycznieniu trybu ich świadczenia, poprawie jakości, obniżce cen oraz dopasowaniu usług do potrzeb użytkowników — uzasadniają przeprowadzenie w dziedzinie usług informatycznych radykalnej i szybkiej reformy. Powinna się ona opierać na kilku zasadniczych, wymienionych poniżej kanonach.

● Podstawową jednostką gospodarczą prowadzącą profesjonalną działalność informatyczną powinno być w pełni samodzielne usługowe przedsiębiorstwo informatyczne. Przedsiębiorstwa takie mogą powstawać w wyniku odpowiedniego przekształcania dotychczasowych jednostek zajmujących się informatyką. Powinny też zostać stworzone prawne podstawy powoływania przedsiębiorstw na zasadzie umów między państwem a twórczymi zespołami profesjonalistów: państwo oddawałoby w dzierżawę majątek produkcyjny (sprzęt i pomieszczenia) oraz udzielałoby koncesji na świadczenie określonego typu usług.

● Przedsiębiorstwo — w ramach koncesji — ma pełną samodzielność w prowadzeniu działalności gospodarczej, przy czym regulacja jego stosunków ekonomicznych z państwem odbywa się tylko poprzez system podatkowy, kredyty bankowe i opłaty dzierżawne. Przedsiębiorstwo (jako osobę prawną) obowiązują oczywiście wszelkie przepisy prawa.

● Fundusze przedsiębiorstwa tworzone są z zysków pochodzących ze sprzedaży świadczonych usług oraz z kredytów bankowych. W przypadku małej firmy należy dopuścić możliwość udziału pracowników lub państwa w kapitale przedsiębiorstwa. Udziały pracownicze zależne byłyby wyłącznie od stażu pracy, a nie od wysokości zarobków. Podział zysku na część akumulowaną i płace jest wewnętrzną sprawą przedsiębiorstwa.

● Wyłączną władzę przedsiębiorstwa stanowi jego samorząd, ustalający — zgodnie z konkretnymi warunkami — wewnętrzną strukturę funkcjonalno-organizacyjną i wewnętrzne zasady zarządzania i administrowania.

● Fundusze akumulowane mogą być przeznaczane zarówno na cele inwestycyjne, jak i finansowanie prac podstawowych i rozwojowych, nie będących bezpośrednio przedmiotem zamówień, lecz mogących później — jako uniwersalne produkty informatyczne — służyć sprzedaży lub ułatwiać świadczenie przyszłych usług. Prace tego typu mogą być też oczywiście zlecane przedsiębiorstwu (np. przez państwo), lecz wówczas musi być określony ostateczny kształt produktu, którego kupna chce dokonać zleceniodawca.

● Jako zasadę stosunków gospodarczych z otoczeniem (z jednostkami gospodarczymi — zleceniodawcami) należy przyjąć posługiwanie się systemem cen negocjowanych, w przypadkach szczególnie prostych, stypizowanych usług informatycznych powinny obowiązywać względnie stałe cenniki.

● Przedsiębiorstwo może się podjąć świadczenia usług kompleksowych, np. z zakresu marketingu, badań statystycznych czy zarządzania, angażując do tego celu fachowych podwykonawców.

● Przedsiębiorstwo może wchodzić w dobrowolne zrzeszenia, mające na celu wspólną realizację większych przedsięwzięć, wspólną politykę eks-

portową, wspólnie wykorzystanie sprzętu, wspólne prowadzenie prac podstawowych i rozwojowych lub wspólne świadczenie bardziej kompleksowych usług (firmy doradztwa organizacyjnego itp.).

● Szczególne znaczenie będzie miała samodzielność przedsiębiorstwa w stosunkach z kontrahentami zagranicznymi. Aczkolwiek jest to swoisty temat tabu, wydaje się, że obawy związane z udzieleniem przedsiębiorstwom pełnych pełnomocnictw w zakresie sprzedaży usług informatycznych za granicą nie są uzasadnione. Na rynku światowym istnieje bowiem duży niezrównoważony popyt na prace projektowo-programistyczne. Ceny, jakie w tym zakresie może zaoferować polski wykonawca są — jak na warunki światowe — wyjątkowo atrakcyjne. Istnieją u nas wolne, nie wykorzystywane możliwości sprzętowe i kadrowe. Jakość sprzętu, aczkolwiek w zasadzie nie pozwala na prowadzenie normalnej eksploatacji systemów informatycznych oraz na efektywną produkcję oprogramowania, umożliwia jednak prowadzenie prac projektowo-programistycznych w o g ó l e. Specjalizacja w tych dziedzinach, w których takie możliwości są wystarczające, a ich wykorzystanie daje skuteczny efekt — jest właśnie szansą eksportu polskiej informatyki. Nie można przy tym nie zauważyć, że tego rodzaju eksport jest jedynym realnym źródłem finansowania rozwoju informatyki. Obecnie eksploatowany sprzęt wymaga jak najszybszej modernizacji. Ewidentny jest jednocześnie kompletny brak środków dewizowych. Muszą one zatem pochodzić z dodatkowego, dotychczas nie realizowanego eksportu. Jak wykazuje z kolei dotychczasowe, niezbyt chlubne doświadczenie w zakresie zakupów sprzętowych, pozostawienie ich w gestii centralnego sponsora może doprowadzić jedynie do spustoszeń. Pozostawienie w tej dziedzinie wolnej ręki przedsiębiorstwom informatycznym stanowi — po pierwsze — pewną gwarancję fachowości przedsięwzięć i tym samym — racjonalizacji importu, a po drugie — daje całkiem realną perspektywę rozwoju bazy sprzętowej. Oddanie do dyspozycji przedsiębiorstwa środków dewizowych przezeń wypracowanych, przy jednoczesnym respektowaniu zasady samofinansowania (a nawet z stosowaniem zasady nie kredytowania przedsiębiorstwa zagranicznymi środkami płatniczymi), nie stanowi dla finansów państwa najmniejszego zagrożenia, tworząc jednocześnie — poprzez podatki — nowe źródło finansowania budżetu środkami dewizowymi.

Wdrożenie tej skrótowo zarysowanej koncepcji samodzielnego przedsiębiorstwa informatycznego wymaga stosunkowo niewielkich przedsięwzięć instytucjonalnych. Jedynym „założycielskim", finansowym wkładem państwa jest rezygnacja z obecnej absurdalnie wysokiej akumulacji sprzętu informatycznego. Jest to bowiem warunek konieczny urealnienia i obniżki cen usług informatycznych.

Radykalizm proponowanej reformy nie wydaje się przesadny i niebezpieczny. Wynika to z wewnętrznej specyfiki jednostek informatycznych. Po pierwsze, fluktuacja kadr jest w tym zawodzie z natury rzeczy stosunkowo niewielka, stąd też sprzeczność między długofalowymi interesami przedsiębiorstwa, a bieżącymi jego załogi jest niejako samoczynnie regulowana (te pierwsze są też bezpośrednio interesami pracowników). Po drugie — zróżnicowanie zarówno produktów przedsiębiorstwa informatycznego, jak i grup zawodowych tworzonych przez pracowników przedsiębiorstwa jest niewielkie. Stąd też żadnego rodzaju „partykularyzmy branżowe" nie stanowią realnego zagrożenia dla jedności firmy i jej wewnętrznej równowagi.

Sądzę, że — rzadko przez historię i warunki gospodarcze stwarzana — szansa nieryzykownego, mogącego przynieść niebagatelne profity ekonomiczne i uprawniającego do wyciągnięcia uogólniających wniosków, eksperymentowania na żywym organizmie społeczno-gospodarczym powinna zostać wykorzystana. W razie niespodziewanych trudności odnowę w informatyce naprawdę łatwo będzie powstrzymać.

Piotr DĄBROWSKI
Komisja Warunków i Organizacji Pracy NSZZ Pracowników Informatyki

POLSKIE TOWARZYSTWO INFORMATYCZNE

Zgodnie z zapowiedzią publikujemy dalsze części referatu programowego, przedstawionego na Zjeździe Założycielskim PTI (na którym referat ten uznano za podstawę działania). W następnym numerze przedstawimy pełny tekst statutu PTI oraz informacje o aktualnych pracach Towarzystwa. (Red.)

Obecny stan informatyki

Informatyka, mimo że wyodrębniona w pracach planistycznych i sprawozdawczych (coroczne opracowanie GUS — „Informatyka i ośrodki informatyczne"), jest organizacyjnie rozczłonkowana pomiędzy prawie wszystkie resorty i urzędy centralne. Powoduje to dość istotne problemy w kształtowaniu strategii rozwoju tej dziedziny. Część Narodowego Planu Społeczno-Gospodarczego dotycząca informatyki jest sumą planów resortów, a nakłady wydatkowane na informatykę są sumą resortowych nakładów przeznaczonych na informatykę. Plan NPSG jest więc budowany na podstawie możliwości finansowych resortów, z uwzględnieniem partykularnych interesów wpływowych grup nacisku, a nie na podstawie potrzeb gospodarki narodowej.

Nakładają się na to takie słabości krajowej gospodarki, jak:

- centralistyczny model sterowania (przy założeniu, że „góra" wie lepiej co jest potrzebne „dołowi")
- wieloszczeblowy system koordynacji (w przypadku informatyki są cztery szczeble), w którym rola koordynatora jest rolą nadzorcy
- brak rzeczywistych (i działających) mechanizmów ekonomicznych
- brak jednolitych mierzalnych kryteriów działalności gospodarczej.

W latach 1975—1979 nakłady na zastosowania informatyki (rozumiane jako suma nakładów inwestycyjnych i kosztów wytworzenia prac i usług informatycznych, pomniejszona o amortyzację) — jako procent produktu narodowego brutto — kształtowały się następująco: 0,74%, 0,71%, 0,65%, 0,68% i 0,69%. Dla porównania przedstawiony jest poniżej ten sam wskaźnik dla Francji i USA. Nakłady inwestycyjne na rozwój zastosowań informatyki w latach 1976—1980 wyniosły 32 mld zł,

	1972	1974	1977	1980
Francja	1,67%	1,87%	2,35%	2,72%
USA	2,51%	2,63%		3,6%

przy czym — począwszy od 1978 r. — systematycznie maleją.

Stan bazy sprzętowej jest niezadowalający pod względem zarówno jakościowym (przestarzałe typy komputerów, niedostateczne konfiguracje, duża zawodność działania), jak i ilościowym. Od 1978 r. zmniejsza się gwałtownie liczba produkowanych komputerów (w 1980 r. było ich 35, a w planie na rok bieżący są tylko 3), nie zapewnia to nawet prostej reprodukcji parku komputerowego, tak że średni wiek komputerów co roku zwiększa się o rok. Podobnie katastro-

falnie wygląda sprawa wyposażenia komputerów w pamięci i sprzęt peryferyjny. Nieliczne tylko egzemplarze mają konfiguracje optymalne, umożliwiające pełne wykorzystanie możliwości komputerów; dotyczy to i serii ODRA, i RIAD.

Sprawa konfiguracji ma znaczenie pierwszoplanowe, gdyż dzięki ich rozbudowie można najmniejszym wysiłkiem inwestycyjnym zwiększyć moc i możliwości obliczeniowe komputerów. Co więcej — ubogie konfiguracje uniemożliwiają stosowanie takich metod dostępu (od bardzo dawna powszechnie używanych na świecie), jak: podział czasu, wielodostęp, zdalny dostęp i praca w sieciach. Ubogie konfiguracje ograniczają możliwość korzystania z oprogramowania firmowego (ICL, IBM). Dostępność oprogramowania jest też hamowana brakiem wyspecjalizowanych przedsiębiorstw, zajmujących się programowaniem, brakiem rzetelnych badań naukowych w dziedzinie zastosowań informatyki oraz brakiem mechanizmów ekonomicznych stymulujących produkcję systemów i pakietów (standardowych i powielarnych). Zespół tych czynników zwiększa czas i koszt wdrażania informatyki w gospodarce.

W organizacji zastosowań dominują dwa typy rozwiązań — przedsiębiorstwo z własnym ośrodkiem obliczeniowym i zlecające obsługę informatyczną na zewnątrz, najczęściej ogólnodostępnym ośrodkom usługowym ZETO lub resortowym ośrodkom usługowym typu ETOB. Niezauważalny jest w tym udział przemysłu komputerowego, który w zbyt małym stopniu zajmuje się produkcją oprogramowania i nie prowadzi prawie wcale akcji doradczych i marketingowych. Nie stosuje się również dzierżawy komputerów, podczas gdy w krajach rozwiniętych działania tego typu są powszechnie prowadzone przez firmy komputerowe.

Współdziałanie informatyka z użytkownikiem informatyki budzi jeszcze ciągle wiele zastrzeżeń. Negatywny wpływ na to współdziałanie ma przede wszystkim fakt, że brak jest wypracowanych metod i technik wprowadzania informatyki do działań gospodarczych. Negatywny wpływ ma również niski stan kultury informatycznej społeczeństwa.

Kształtowanie cen sprzętu informatycznego, oprogramowania i usług informatycznych (wysoki podatek w postaci akumulacji, cennik usług oparty na relacjach godzinowych) oraz istnienie rynku producenta środków i usług informatyki, a przy tym niskie koszty pracy żywej — wszystko to powoduje, że trudno jest mówić o jakiejkolwiek ekonomice informatyki. Do tego dochodzi przekonanie, że za pomocą stosowanych obecnie kryteriów ocen można łatwo wykazywać przydatność informatyki dla gospodarki i społeczeństwa, przekonanie, powodujące często ekwilibrystyczne obliczenia oparte na pseudoekonomicznym rachunku. Jest to jeszcze jeden rezultat braku ekonomicznych mechanizmów, stymulujących stosowanie i rozwój informatyki.

Do czynników ograniczających rozwój zastosowań należą braki oraz zła jakość materiałów eksploatacyjnych (papierowe i magnetyczne nośniki informacji). Produkcja papierowych nośników informacji jest ulokowana w resorcie leśnictwa co — w obliczu kłopotów tego resortu — oznacza właściwie przerwanie produkcji tych materiałów. Przy tym produkcja urządzeń końcowych i systemów gromadzenia danych ograniczających i eliminujących stosowanie papierowych nośników informacji jest jeszcze ciągle za mała.

Niedostateczny jest również stan prac normalizacyjnych w dziedzinie informatyki.

Bardzo istotne są ograniczenia możliwości rozwojowych kadry. Brak stopni specjalizacyjnych, zdeprecjonowanie stanowiska tzw. specjalistów oraz niejednolitość i na ogół niski poziom wymagań egzaminacyjnych na różnych kursach informatycznych — wszystko to uniemożliwia wprowadzanie jakichkolwiek kryteriów kwalifikacyjnych przy obsadzaniu stanowisk „informatycznych". Poszczególne grupy zawodowe w środowisku informatyków popadają w swego rodzaju zachowawczość rozwojową. Zaniedbywane są problemy jakości sprzętu i zastosowań, przy jednoczesnej fetyszyzacji pewnych typów maszyn.

Wyższe uczelnie opuszcza rocznie ok. 500 absolwentów o specjalności „informatyka", dość wszechstronnie wykształconych pod względem naukowym, ale nie mających dostatecznego obycia ze sprzętem. Wynika to z braku nowoczesnych laboratoriów komputerowych oraz ogólnie niewystarczającego (i relatywnie coraz gorszego) wyposażenia uczelni w sprzęt informatyczny. Szerzy się praktyka zbywania ogólnikami tych elementów edukacji informatyków, które wymagają praktycznych umiejętności programowania i pracy z komputerem.

Konflikt pokoleń w przypadku dość młodej informatyki nie nabrał jeszcze ostrości, ale w sytuacji ogólnogospodarczego kryzysu przed nowo promowanymi informatykami z wyższym wykształceniem rysuje się groźba braku zatrudnienia. Jakiekolwiek jednak tendencje do drastycznego ograniczenia kształcenia informatyków mogłyby w skali perspektywicznej doprowadzić do nieobliczalnych skutków ujemnych i dalszego pogłębienia naszego informatycznego zacofania.

Na tle ogólnej sytuacji pewnym osiągnięciem obserwowanym w ciągu ostatnich lat, a wynikającym z „choroby ubóstwa" — jest fakt, że ośrodki obliczeniowe coraz większą uwagę zwracają na rozbudowę konfiguracji zainstalowanych systemów komputerowych i bardziej efektywne ich wykorzystanie. Następuje wyraźne porządkowanie posiadanych zasobów. Są też podejmowane próby standaryzacji oprogramowania i osiągnięcia jego powielarności, mimo braku mechanizmów ekonomicznych stymulujących ten kierunek działań.

Odczucia społeczne dotyczące potrzeby stosowania informatyki w gospodarce są zróżnicowane. Wyrażane są opinie — od nadmiernie entuzjastycznych (informatyka czynnikiem postępu), przez pozornie obiektywne (informatyka wzmacniaczem nonsensu), aż do nihilistycznych (i po co to wszystko?). Na taką rozbieżność stanowisk wpływają — obok rzeczywistych słabości informatyki — brak kontaktu, nawet pośredniego, przeważającej części społeczeństwa z informatyką (np. w bankach, PKO, na poczcie) oraz prymitywne i naiwne metody propagowania informatyki przez środki masowego przekazu. Zrażona część społeczeństwa przekonana jest, że Polska — a w szczególności jej gospodarka — może znakomicie egzystować bez informatyki (aczkolwiek wiele osób, z obawy przed posądzeniem o konserwatyzm, głośno się do tego nie przyznaje). Ci natomiast, którzy mieli pozytywny kontakt z informatyką jako narzędziem w swojej pracy, nie negują jej, przeciwnie — jej rozwój uważają za konieczny.

Do czego dążyć w polskiej informatyce?

Każdy współczesny system społeczno-ekonomiczny, który pragnie zapewnić narodowi rozwój oświaty i kultury oraz dostarczyć mu dóbr materialnych, powinien w kierowaniu państwem, zarządzaniu gospodarką, sterowaniu procesami produkcyjnymi itp. korzystać z aktualnej i rzetelnie opracowanej informacji. Tak więc i w naszym kraju informatyka musi stać się wreszcie towarem wartościowym i atrakcyjnym, którego cena zależeć będzie od jej wiarygodności, jak i czasu udostępnienia. Gromadzenie informacji, jej przetwarzanie i przekazywanie w odpowiednio krótkim czasie wymaga już dzisiaj posługiwania się informatyką.

Korzyści wynikające z zastosowania informatyki można sklasyfikować następująco:

a) korzyści wynikające z prowadzenia działalności wytwórczej w zakresie sprzętu informatycznego i oprogramowania
b) bezpośrednie korzyści ekonomiczne wynikające z zastąpienia pracy ludzkiej pracą komputerów
c) rozwój wielu dziedzin nauki i techniki, często niemożliwy bez środków i metod informatyki
d) pośrednie korzyści ekonomiczne i organizacyjne wynikające z szybszego i pełniejszego dostarczania informacji niezbędnej w procesach zarządzania, sterowania produkcją oraz w usługach; w tym za-

kresie informatyka może się przyczynić do znacznego wzrostu dochodu narodowego i optymalizacji struktur organizacyjnych
e) rozwój cywilizacyjny przez wzrost kultury informatycznej społeczeństwa.

Umiarkowany optymizm pozwala zakładać, że w obecnych i dających się przewidzieć warunkach polskich:

ad. a) przemysł komputerowy powinien zaspokoić krajowe potrzeby w odniesieniu do sprzętu i oprogramowania z taką nadwyżką, która umożliwiłaby import takich urządzeń i systemów, których wytwarzanie w kraju nie jest uzasadnione — dając w łącznych obrotach z zagranicą saldo zerowe; należy krytycznie przeanalizować opłacalność produkcji komputerów w Polsce, a zwłaszcza ich eksportu (wskazany byłby rozwój niewielkich systemów komputerowych, konstruowanych na bazie mikroprocesorów, zwłaszcza zaś w zakresie oprogramowania);

ad. b) nie należy się spodziewać większych wyników, a to ze względu na rażące dysproporcje pomiędzy cenami pracy ludzkiej i sprzętu komputerowego;

ad. c) trzeba się szczególnie zająć zastosowaniem mikroprocesorów we współczesnej aparaturze pomiarowej, automatyce i biomedycynie, a także zastosowaniami optymalizującymi i stymulacyjnymi w różnych dziedzinach;

ad. d) należałoby udzielić preferencji informatycznemu systemowi dla bankowości, a także systemom informacji naukowo-technicznej i ekonomicznej; pozostałe decyzje o komputeryzacji powinny być scedowane na szczebel przedsiębiorstwa;

ad. e) należy dążyć do upowszechnienia wiedzy informatycznej i umiejętności posługiwania się środkami informatyki, zwłaszcza wśród młodzieży.

Jednym z podstawowych warunków jest w tym przypadku doprowadzenie do pozytywnej selekcji w zawodzie informatyka i stworzenie — obok awansu „pionowego" — możliwości atrakcyjnego awansu „poziomego". Podstawowym warunkiem zewnętrznym jest doprowadzenie do sytuacji, w której istniałoby zapotrzebowanie na informację. Zadania związane z poszczególnymi przedsięwzięciami informatycznym powinny być zalecane imiennie osobom, mającym wolną rękę w sprawach kadrowych i w realizacji przedsięwzięcia (w granicach przyznanych środków).

Po uwzględnieniu wszystkich istniejących dzisiaj ograniczeń będzie możliwe opracowanie pożądanych wariantów rozwoju. Na przykład — załamanie się globalnych „informatycznych systemów rządowych" sugeruję, że przy niedorozwoju struktur organizacyjnych i braku obiektywnego zapotrzebowania na rzetelną informację należy rozważyć przejściowe ograniczenie rozwoju systemów na szczeblu centralnym. Należy przypuszczać, że w związku z kierunkami reformy gospodarczej większe proporcjonalnie środki przeznaczone będą na doskonalenie i rozwój systemów na szczeblu przedsiębiorstw, a zwłaszcza banków.

Mamy dość dużo całkiem udanych rozwiązań, które nie są w dostateczny sposób rozreklamowane i wykorzystane (jak choćby systemy optymalizacji transportu). Absurdalne przepisy i działania wywodzące się z nieracjonalnych zasad gospodarowania powodowały często, że nawet udane systemy okazywały się nieprzydatne w praktyce i były odkładane na półki. Podobne skutki wynikały niejednokrotnie z obawy przed ujawnieniem źródeł wadliwego działania struktur produkcyjnych i organizacyjnych.

Wydaje się, że wiele systemów informatycznych opracowano jedynie z punktu widzenia potrzeb informatyków, poszukujących eleganckich lub naturalnych zastosowań dla znanych sobie narzędzi. Sukcesy uzyskiwano natomiast często w zastosowaniach, które — choć im wystarczały bardzo proste metody — wymagały doskonałej znajomości informatyzowanych problemów. Z kolei niedokształceni informatycznie przedstawiciele wszystkich dziedzin zastosowań tworzyli systemy chałupnicze — z informatycznego punktu widzenia — i mało uniwersalne, które nie mogły być rozpowszechniane. Obecnie niezbędny jest rozwój zastosowań w dziedzinie usług, rolnictwa, ochrony zdrowia i opieki społecznej oraz systemów informacyjnych i bankowych (przy aktywnym współudziale specjalistów-nieinformatyków, o dużym doświadczeniu w danych dziedzinach).

Należy skończyć z fascynacją sprzętem, gdyż na świecie jego udział w globalnej wartości systemów informatycznych gwałtownie się zmniejszył i nadal będzie malał. Preferencje powinny być udzielane przede wszystkim rozwojowi oprogramowania, aby uniknąć sytuacji, w której powstaje ono dopiero po rozpoczęciu produkcji sprzętu. Ponadto wyniki prac programistycznych mogą stanowić racjonalną, atrakcyjną ofertę eksportową.

Opracowanie koniecznych przedsięwzięć dla przegrupowania sił i środków jest niemożliwe bez współdziału i akceptacji środowiska informatycznego. Środowisko to powinno być inicjatorem szczegółowych przedsięwzięć informatycznych i wskazywać najbardziej korzystne obszary zastosowań. PTI, instytucjonalny przedstawiciel środowiska, powinno przede wszystkim dostarczać ocen jednoznacznie wartościujących ważniejsze przedsięwzięcia informatyczne — zrealizowane i zamierzone.

Cele Polskiego Towarzystwa Informatycznego

Środowisko informatyczne jest bardzo rozproszone. Ogólnym celem PTI jest więc konsolidacja środowiska informatyków i ukierunkowanie ich działań na rozwój działalności naukowej i efektywne wykorzystanie środków informatyki — z punktu widzenia potrzeb społecznych i gospodarczych. Zadaniem Towarzystwa jest kształtowanie *esprit de corps* informatyków, wypracowanie wzorców etyki zawodowej oraz czuwanie nad ich przestrzeganiem, dbanie o sumienny i uczciwy oraz pozbawiony frazesów i ogólników styl publikowania i przedstawiania prac, a także popieranie działań społecznych w kierunku zachowania sfer życia prywatnego, nie objętych wielkimi systemami informacji personalnej.

Celem Towarzystwa jest stworzenie form wyrażania i kształtowania profesjonalnych opinii informatyków, a także reprezentowanie ich opinii, potrzeb i uprawnień wobec społeczeństwa, władz oraz stowarzyszeń w kraju i za granicą. Opinie wypracowane w środowisku będą prezentowane na zewnątrz głównie drogą publikacji w literaturze fachowej. Jedną z form prezentowania profesjonalnych opinii powinny być także seminaria tematyczne.

W związku z krytyczną oceną dotychczasowych efektów stosowania informatyki, PTI podejmie się w pierwszej kolejności analizy przyczyn tego stanu rzeczy, podejmie szerokie współdziałanie z zainteresowanymi jego zmianą. Dla ustanowienia właściwej rangi informatyki w społeczeństwie PTI będzie krzewiło wiedzę o efektach, jakie daje lub może dać informatyka, oraz o tym, gdzie i jak ją stosować. Pozwoli to na uświadomienie społeczeństwu celowości stosowania informatyki przy czekających nas zmianach w gospodarce.

PTI będzie inicjowało i rozwijało przedsięwzięcia zmierzające do prawidłowego wykorzystania i rozwoju informatyki i informatyków. Szczególnie ważne będzie współdziałanie z przemysłem przy projektowaniu i wdrażaniu środków informatyki. Towarzystwo zajmie się popularyzacją współczesnych technologii produkcji oprogramowania oraz będzie współdziałać przy ich opracowywaniu i stosowaniu. Będzie się starało wyrażać opinie o zapotrzebowaniu na poszczególne typy urządzeń informatycznych (zwłaszcza peryferyjnych) i ich przydatności.

PTI będzie poszukiwać nowoczesnych metod prowadzenia prac związanych z informatyką oraz je popularyzować. Będzie podejmować działania zmierzające do wzbogacenia wiedzy specjalistów z różnych dziedzin, dla których informatyka mogłaby być cennym narzędziem, a także wiedzy informatyków.

PTI będzie poszukiwało form organizacyjnych pozwalających na zwiększenie skuteczności działania informatyki i dążyło do zapewnienia rzetelnych ekspertyz informatycznych. Ponadto Towarzystwo prowadzić będzie działalność popularyzatorską.

zjednoczenie informatyki



BAZY DANYCH – konieczność czy sztuka dla sztuki?

Na początku winien jestem Czytelnikom wyjaśnienie, że poruszana w niniejszym artykule tematyka stanowi kontynuację problematyki systemów zarządzania bazami danych (SZBD), przedstawionej w krótkim artykule mgr inż. Jerzego Pasuli, zamieszczonego w INFORMATYCE nr 2/81, a dyskutowanej podczas III Międzynarodowego Seminarium w Zaborowie k. Warszawy (listopad ub.r.). Wyjaśnienie to jest konieczne, gdyż w aktualnej, trudnej sytuacji gospodarczej kraju, a w szczególności sytuacji zaopatrzeniowej, która w ostatnich tygodniach nęka również wiele ośrodków obliczeniowych (np. zdobycie papieru do drukarek lub kart maszynowych stanowi obecnie nie lada problem), rozpatrywanie tematyki SZBD może wydawać się nie na czasie.

Jest jednak kilka istotnych powodów, które skłoniły mnie do zabrania głosu na ten temat. Pierwszy — to wpływ baz danych i systemów zarządzających nimi (SZBD) na dalszy rozwój informatyki. Ich roli nie negowali w trakcie seminarium ani specjaliści z państw zachodnich (Austria, Finlandia, RFN), ani ze wschodnich (Bułgaria, CSRS, NRD, Węgry, Polska).

Szczególne zainteresowanie bazami danych jakie wykazywali specjaliści z niektórych państw socjalistycznych — i to zarówno podstawami teoretycznymi (modelowymi), jak i ich budową (z uwzględnieniem własnych oryginalnych rozwiązań programowych w zakresie SZBD) wynika m.in. z faktu, że komputery Jednolitego Systemu dostarczane są przez poszczególnych producentów z ubogim jeszcze oprogramowaniem — podstawowym (narzędziowym) i użytkowym.

W krajach zachodnioeuropejskich natomiast potencjalny użytkownik bazy danych może zakupić potrzebne mu oprogramowanie od producenta systemu komputerowego. Ta korzystna z pozoru sytuacja użytkownika stwarza jednak pewne niebezpieczeństwo, rzadko u nas dostrzegane. Przyszły użytkownik bazy danych, nie będąc zainteresowanym produkcją oprogramowania narzędziowego, zdany jest przy ocenie optymalności rozwiązania bazy danych bądź na opinię producenta sprzętu, bądź na zdanie specjalnie zatrudnionego konsultanta. Praktyka przy tym wskazuje, że najczęściej decydują tu względy czysto handlowe, a nie wiedza merytoryczna, poparta konieczną analizą.

Stwarza to — jak wykazały niektóre referaty i seminaryjna dyskusja — taką sytuację, że w wielu krajach zachodnioeuropejskich (poza nielicznymi ośrodkami uniwersyteckimi oraz firmami komputerowymi, zainteresowanymi głównie rozwiązaniami aplikacyjnymi o możliwie największym stopniu uniwersalności) nie prowadzi się poważnych studiów nad bazami danych. Stąd wynika zainteresowanie użytkowników z państw zachodnich konkretnymi opracowaniami wschodnioeuropejskimi — szczególnie tych użytkowników, dla których uniwersalność pakietów oprogramowania systemowego baz danych z wielkich firm komputerowych stanowi niepotrzebny „balast", podwyższający nadmiernie koszt inwestycji.

Potwierdzeniem powyższych rozważań był między innymi obszerny referat K. Suppera [8], przedstawiający wyniki studium, (opracowanego przez międzynarodowe kolegium), które dotyczy sposobów użytkowania baz danych przez 120 różnych użytkowników z czterech krajów zachodnioeuropejskich (Anglia, Francja, RFN, Włochy). To unikalne studium przedstawia wyczerpujące trudności, jakie wiążą się z implementacją oraz eksploatacją baz danych.

Argumentem dla tych rozważań były również wystąpienia specjalistów wschodnioeuropejskich prezentujące takie chociażby rozwiązania, jak: SZBD „DAFEMA" [2], realizowany na komputerze NRD R-40 (dla różnych użytkowników nie dysponujących technicznymi możliwościami eksploatacji SZBO oferowanych przez duże firmy komputerowe) czy SZBD „RODAN" (CPiZI), który — jako oryginalne polskie rozwiązanie — wdrożony został na kilkudziesięciu (ok. 30) instalacjach maszyn RIAD.

Szczególnie interesująco zaprezentował się dr W. Staniszkis, który — zajmując się problemami wspomaganego komputerowo projektowania SZBD — przedstawił charakterystykę nowo opracowanego symulatora bazy danych [7]. Jego zdaniem kompleksowość decyzji projektowych, konsekwencje w zakresie kosztów, jak i możliwość powstawania błędów, wymuszają zastosowanie narzędzi komputerowych w projektowaniu baz danych. Z kolei M. Kowalewski [4] podał nowe możliwości RODAN, który po opracowaniu zaprojektowanego pakietu SMI (Standard Minicomputer Interface) będzie mógł zarządzać sieciami minikomputerów. Przedstawiono przy tym kompletny protokół aplikacyjny współpracy minikomputerów, uwzględniający między innymi obsługę programów działających w trybie konwersacyjnym.

Drugi powód, obligujący mnie do ponownego poruszenia spraw związanych z tematyką Seminarium, to przedstawiony tam stan zastosowań w zakresie SZBO w poszczególnych krajach. Stwierdzono, że prace związane z projektowaniem i zastosowaniem baz danych są przedsięwzięciami wymagającymi dużego nakładu pracy i kosztów, w związku z czym należy przeprowadzić dokładną analizę dotychczasowych krajowych rozwiązań pod kątem efektywnego ich zużytkowania. W przeciwnym razie musielibyśmy pogodzić się z nieuzasadnioną w wielu przypadkach kontynuacją prymitywnego wykorzystania posiadanego w kraju potencjału komputerowego, nie mówiąc już o modelowaniu zamierzeń rozwojowych. Poza tym nieprzychylna atmosfera, która wytworzyła się wokół informatyki — uzasadniona poniekąd niewielkimi efektami ekonomicznymi uzyskanymi dotychczas — może (przy naszej skłonności do popadania w skrajność) doprowadzić do sytuacji, w której reforma gospodarki narodowej prowadzona będzie za pomocą liczydeł.

Przyjrzyjmy się zatem zastosowaniom SZBD zaprezentowanym na Seminarium.

I. Bana z DATAORG (Węgry) przedstawił koncepcję bazy danych opracowaną dla Ministerstwa Handlu Zagranicznego, w której rejestrowane są dane dotyczące ponad miliona kontraktów zawieranych w ciągu roku przez węgierskie centrale handlu zagranicznego. Do zarządzania bazą wykorzystano pakiet firmy SIEMENS — UDS. W projektowaniu bazy zastosowano dwie metody: logiczną i fizyczną. Pierwsza — wspomagana jest maszynowo, niezależnie od faktycznej struktury danych, ukształtowanej jeszcze w latach, gdy przetwarzanie odbywało się bez bazy danych i na innym komputerze. Po zakończeniu opracowania struktury logicznej, nastąpiło jej dopasowanie, zarówno do nowego typu komputera, jak i wyżej wspomnianego systemu zarządzania bazą.

Dr B. Freyer omówił bazę danych dla centralnego dostawcy materiałów budowlanych w NRD, zawierającą dane zebrane z 300 magazynów. Materiały budowlane dostarczane są wg

centralnego rozdzielnika, co kontrolowane jest przez komputer. Każdego dnia dane, dotyczące ok. 50 tys. różnych tras transportu materiałów, rejestrowane są na taśmie magnetycznej i przekazywane do centralnego ośrodka. Ośrodek ten wyposażony jest w komputer R-40. W procesie przetwarzania uczestniczy 600 stacji wprowadzania danych oraz 14 ośrodków regionalnych, wyposażonych w komputery II generacji R-300. Obliczeniami objęte są takie dziedziny działalności, jak: fakturowanie (1500 faktur dziennie), rozliczenia bieżące, materiałowe i kosztów. Opracowywana jest pełna statystyka.

J. Kovać i M. Sojakowa przedstawili bazę danych użytkowanych przez koncern SLOVCHEMIA z CSSR. Koncern ten składa się z 25 jednostek: 17 fabryk, 2 central handlu zagranicznego oraz 6 instytutów badawczych. Wykonuje on ok. 1/3 całej produkcji przemysłu chemicznego CSSR. Do zaprojektowania bazy danych przystąpiono po ok. 10 latach wsadowego przetwarzania na komputerze IBM 360/30. Budowę bazy rozpoczęto przed dwoma laty po zakupieniu komputera IBM 370/148. Przyjęto hierarchiczną strukturę AMS (Automated Management System) z elementami zdalnego, przestrzennego rozproszenia. Dla celów zarządzania koncernu baza dostarcza dane dotyczące „planowania" (alternatywne prognozy produkcyjne), „regulacji" (eliminowanie sytuacji awaryjnych w produkcji) i „kontroli" (wielkość wykonanej produkcji, statystyki oraz analizy).

S. Kiss z Węgier zaprezentował, będącą obecnie w trakcie dalszych opracowań, bazę danych dla VOLAN TRUST, państwowego koncernu transportowego, obsługującego całe Węgry i zagranicę. W projekcie przewidziano zbudowanie sieci minikomputerowej oraz jej obsługę poprzez rozproszoną bazę danych o nazwie VDDB (Volan Distributed Data Base). Zastosowane zostały minikomputery produkcji węgierskiej (typ TPA-11/40) oraz sprawdzone oprogramowanie w postaci pakietów DOSNET i PDP—DBMS. Aktualnie instaluje się pierwsze trzy minikomputery, które mają wykorzystać wspomagane oprogramowanie.

Z problemami rozproszonych baz danych, ale w aspekcie bardziej ogólnym, zapoznał słuchaczy P. Kaiser [3]. Jego zdaniem częste w ostatnim okresie dyskusje specjalistów na temat przetwarzania rozproszonego wynikają z faktu, że w następstwie doświadczeń zebranych w trakcie eksploatacji wielkich scentralizowanych systemów określone zostały dla nich nowe wymagania. Poza tym intensywny rozwój sprzętu komputerowego i oprogramowania (rozwój technik mikroprocesorowych poprzez LSI i VLSI oraz setki przeróżnych zastosowań SZBD i oprogramowania komunikacyjnego), stwarza zupełnie nowe możliwości realizacyjne.

Ostatnie jednak opinie, wynikające z aktualnie prowadzonych prac, wskazują na to, że problemy związane z projektowaniem i wdrażaniem systemów zarządzania dla rozproszonych baz danych nie są proste i jak dotychczas nie zostały w sposób zadowalający rozwiązane. I chociaż uzyskane obecnie rezultaty pozwalają przypuszczać, że rozproszone bazy danych będą posiadać pewną przewagę nad systemami scentralizowanymi (mniejsze wymagania w zakresie wielkości pamięci, większą niezawodność i podatność na adaptacje, czy mniejsze koszty transmisji...), dopiero przyszłość pokaże, czy oczekiwania zostaną spełnione.

A. Baczko zapoznał z kolei słuchaczy z koncepcjami symulatorów oraz ich zastosowaniem w różnych strukturach rozproszonych baz danych [1]. Przedstawione przez niego metody, odnoszące się do sieci komputerowych, mogą znaleźć praktyczne zastosowanie w projektowaniu SZBD, a w szczególności systemu zarządzania rozproszoną bazą danych.

Oryginalną koncepcję zaprezentował J. Pasula w referacie [6], w którym przedstawił strukturę architektoniczną „Data Base Machine" (DBM) dla CODASYL-owskiego systemu zarządzania bazą. Zdaniem autora — dzięki zbudowaniu DBM można oczekiwać uzyskania wielu nowych możliwości, rozszerzających dotychczasowy zakres zastosowań, oraz uproszczenia funkcji SZBD.

Odrębną tematyką zajął się K. Nindel]5[, który w wyniku przeprowadzonej analizy doszedł do wniosku, że często używany relacyjny język zapytań SEQUEL-2 nie jest w pełni przydatny w operacjach wyszukiwania dla systemów informacyjnych opartych na bazie danych, a w szczególności — systemu AIDOS, opracowanego przez firmę ROBOTRON z NRD. Wynika to z przekonania autora, że język SEQUEL — bardzo użyteczny dla standardowych zapytań, dotyczących typowych dokumentów — raczej nie nadaje się do obsługi zapytań związanych z semantycznymi powiązaniami, dla których lepszą prezentację stwarza model semantyczny, np. model Roussopoulosa.

W dyskusji wielu mówców potwierdziło istnienie rozwierającej się luki między wymaganiami użytkowników a pracami rozwojowymi w zakresie oprogramowania, związanymi z konkretnymi zastosowaniami. Kilku dyskutantów wyraziło również wątpliwość, czy dalszy szeroki rozwój prac teoretycznych jest właściwym kierunkiem (wobec naszych ogólnie znanych trudności ze sprzętem), który umożliwiałby wykorzystanie tych prac. Z drugiej zaś strony J. Pasula, rozwijając swoje tezy podane w referacie [6], był zdania, że np. zaprzestanie dalszych prac nad „Data Base Machine" może spowodować zaprzepaszczenie obecnej szansy zbliżenia poziomu prac specjalistów Wschodu i Zachodu. Niejako w odpowiedzi, prof. A. Berger z Austrii wyraził pogląd, że prace teoretyczne zawsze rozwijały się niezależnie od praktycznych zastosowań. Nawet jeżeli opracowana teoria nie ma obecnie szans realizacji, to jednak w wyniku prac badawczych powstają idee, które mogą doprowadzić do opracowania zupełnie nowych koncepcji, spełniających teoretyczne założenia. Dobry przykład stanowi idea DBM, która może spowodować skonstruowanie nowej generacji maszyn cyfrowych.

Zdaniem wielu dyskutantów, przyszłość problematyki baz danych widziana w dziesięcioletniej perspektywie — to rozwiązania sprzętowe i programowe związane z konkretnymi zastosowaniami. Zmieni to w zasadniczy sposób obecnie stosowaną metodologię badawczą oraz funkcje SZBD. Konieczne jest kontynuowanie prac nad tymi systemami. Dyskusję, czy jest to działalność pożyteczna dla rozwoju krajowej informatyki, czy — jak to niektórzy malkontenci mniemają — uprawianie „sztuki dla sztuki" pozostawiam Czytelnikom do rozstrzygnięcia.

LITERATURA (referaty przedstawione na Seminarium):

[1] Baczko A., PAN/PL: Simulation control of distributed data base

[2] Insensee E., DVZ Magdeburg: Data Bank System DAFEMA base for applications of remonte data processing with mean producitivity

[3] Kaiser P., CRC/CSSR: Distributed data bases — some problems and questions

[4] Kowalewski M., CPIZI/PL: Standard minicomputer interface in RODAN transaction processing

[5] Nindel K., LFA/NRD: Formalization of information retrieval system by means of CODD's relational model-possibilities and restrictions

[6] Pasula J., CPIZI/PL: A CODASYL-type DBMS as a data base machine

[7] Staniszkis W., CPIZI/PL: The data base simulatora tool for the computer aided data base design

[8] Supper K., MBP/Gmb H/RFN: How to apply data base in the proper way. An empirical study of current data base software usage.

Józef PIASECKI
ZETO Olsztyn

MEDICAL INFORMATICS EUROPE '81

W połowie lat siedemdziesiątych dwanaście krajowych stowarzyszeń zajmujących się problematyką informatyki medycznej utworzyło organizację pn. European Federation for Medical Informatics (EFMI). W skład tej federacji wchodzą aktualnie następujące kraje: Belgia, Dania, Finlandia, Francja, Hiszpania, Holandia, Norwegia, RFN, Szwecja, Wielka Brytania oraz Włochy. Federacja rozporządza własnym periodykiem naukowym — kwartalnikiem „Medical Informatics" — wydawanym od 1976 roku w Anglii. Jednym z najbardziej istotnych przejawów działalności federacji EFMI są kongresy pn. MEDICAL INFORMATICS EUROPE (MIE).

Pierwszy Kongres MIE'78 odbył się w 1978 r. w Cambridge, drugi — w 1979 r. w Berlinie Zachodnim.

Trzeci z kolei kongres odbył się w Toulouse w dniach od 9 do 13 marca 1981 r. Jego organizatorami były: AIM (L'Association pour l'Application de l'Informatique á la Medicine) i INRIA (L'Institut National de Recherche en Informatique et en Automatique). Przewodniczącym Komitetu Naukowego kongresu był prof. Francis Gremy z Paryża.

Program kongresu obejmował — podobnie jak w Berlinie — trzy generalne tematy:

- **systemy opieki zdrowotnej:** analiza, ocena i planowanie
- **choroby przewlekłe:** prowadzenie, ocena typu followup i badania kliniczne
- **osoby niepełnosprawne:** wspomaganie informatyczne i robotyczne.

W ramach powyższych tematów odbyto sesje robocze. Trzeba jednak stwierdzić, że w odczuciu wielu uczestników kongresu — w tym i niżej podpisanego — sformułowanie tematów wiodących były zorientowane nadmiernie w stronę medycyny.

Szczególnie wartościowymi elementami programu kongresu były sesje szkoleniowe (teaching sesions), posiedzenia tzw. okrągłego stołu oraz niektóre referaty plenarne. Można tu przykładowo wymienić:

- wykład K. Sautera (Kolonia, RFN) „Medyczne bazy danych — podstawowe notacje o strukturze i użytkowaniu"
- dyskusję okrągłego stołu nt. baz danych w naukowych badaniach medycznych
- referaty plenarne:

— J. R. Moehra (RFN) „Komputer w gabinecie lekarskim"

— J. Vidal (USA) „Zintegrowane przetwarzanie danych i telekomunikacja w badaniach biomedycznych i medycynie — perspektywy na lata osiemdziesiąte"

— M. Goldberg, W. Daba i F. Gremy „Miara zdrowia jednostki a miara zdrowia populacji".

W programie znalazły się dwie polskie prace. W ramach sesji „Motor defects" prof. A. Morecki (Politechnika Warszawska) wygłosił referat ilustrowany filmem pt. „Substituting of upper human extermities functions — where are we going? (współautorzy: H. Borowski, J. Kiwerski, R. Paśniczek, J. Ober). Dr P. J. Jasiński (Politechnika Poznańska) przedstawił w sesji „Data handling problems" pracę pt. „Relational data base CARDIOCOD: a tool for clinical research in cardiology" (współautorzy: H. Krug, B. Talkowska). Pełne materiały kongresowe, tj. teksty ponad 100 referatów, zostały wydane przez Springer-Verlag, jako tom 11 serii Lecture Notes in Medical Informatics.

W powszechnym odczuciu uczestników (a było ich ponad 300) kongres MIE był udany tak z punktu widzenia naukowego, jak i organizacyjnego. Następny kongres MIE odbędzie się w marcu 1982 roku w Dublinie.

* * *

Na tle tego krótkiego omówienia federacji MIE i jej kongresów warto pokusić się o garść refleksji dotyczących znaczenia i perspektyw informatyki medycznej.

Po pierwsze — wydaje się, że medyczne zastosowania informatyki definitywnie wyszły z okresu pionierskiego. Wiele projektów komputeryzacji medycyny ma już wieloletnią historię, a w codziennej praktyce klinicznej bądź szpitalnej dowodzi swej użyteczności, rozważanej łącznie z kosztami, które przy takich zamierzeniach trzeba ponosić. Wyciąganie jednak wniosku, iż metody i środki informatyki stały się powszechnym narzędziem lekarskim byłoby błędem. Na przykład, w tak zaawansowanym informatycznie kraju jak USA w większości szpitali bądź nie podjęto prac, bądź też wręcz wyraża się brak zainteresowania komputerowo-wspomaganymi Medycznymi Systemami Informacyjnymi (tzw. MIS). Wiele badań z tego zakresu, prowadzonych w ośrodkach zachodnich, ma do dziś charakter wycinkowy i eksperymentalny; dość często uzyskiwane rezultaty nie spełniają wstępnych założeń, stanowiąc źródło frustracji i dezaprobaty użytkowników, tj. lekarzy bądź administratorów służby zdrowia.

Po drugie — można zaryzykować twierdzenie, że szereg polskich badań i prac z zakresu informatyki medycznej (głównie te, które znane są autorowi niniejszego omówienia z problemu węzłowego PAN 10.4, prowadzonego w latach 1976—1980) reprezentuje poziom w pełni odpowiadający aktualnemu stanowi w Europie, a także na świecie. Niestety, refleksja ta dotyczy głównie sfery koncepcyjnej rodzimych prac; z dobrze znanych powodów nasze realizacje są na ogół podejmowane w ograniczonych warunkach dostępności i jakości sprzętu, co przekreśla ich konkurencyjność.

Można postawić pytanie, czy w obecnej sytuacji naszego kraju wprowadzenie komputerów do medycyny zasługuje na dalszą promocję. Przy zachowaniu należnego umiaru w określaniu celów, odpowiedź na takie pytanie winna być pozytywna i to co najmniej z dwóch powodów. Po pierwsze: zaniedbanie tej dziedziny badań naukowych i działalności projektowej, poniechanie wykorzystania dotychczasowych wyników może się okazać za lat kilka stratą nie do odrobienia. Po drugie: ochronie zdrowia w Polsce od dawna przypisuje się wysoką rangę społeczną (ten aspekt systemu ochrony zdrowia jest także coraz silniej akcentowany w rozwiniętych krajach kapitalistycznych). Trzeba sobie w związku z tym uświadomić, że objęcie wielomilionowej populacji skutecznym, sprawnym systemem ochrony zdrowia, zapewnienie każdej jednostce pełnego dostępu do świadczeń i zadowalającego poziomu obsługi medycznej oraz optymalizacja ogromnych kosztów inwestycyjnych i eksploatacyjnych takiego systemu będą niemożliwe bez wprowadzenia różnorodnych metod i techniki komputeryzacji.

Piotr J. JASIŃSKI

Algorytmy '81

ALGORYTMY to nazwa międzynarodowej konferencji poświęconej prezentacji zarówno nowych algorytmów, jak i nowych metod zastosowania informatyki w gospodarce, technice i nauce. Od 1971 r. co dwa lata w Štrbske Pleso (słowacka część Wysokich Tatr) spotyka się około 300 specjalistów czechosłowackich, reprezentujących szkoły wyższe, instytuty naukowe oraz jednostki organizacyjne przemysłu i administracji. Konferencja organizowana jest przez stały Komitet Programowy z udziałem Słowackiej Akademii Nauk, Towarzystwa Słowackich Matematyków i Fizyków oraz Politechniki w Bratysławie.

Dzięki konsekwencji organizatorów w zachowaniu przyjętej od początku problematyki i towarzyszącym konferencji wydawnictwom cieszy się ona dużym zainteresowaniem i popularnością nie tylko w CSRS, ale również w sąsiadujących krajach (PRL, NRD, ZSRR). Głównym jej celem jest upowszechnienie osiągnięć i wiedzy w dziedzinie metod tworzenia i stosowania algorytmów oraz wymiana doświadczeń w tym zakresie.

Szczegółowy program VI Międzynarodowej Konferencji ALGORYTMY'81 (6—10.IV.1981 r.) obejmował **57 referatów**, wygłoszonych na posiedzeniach sesyjnych (w dwu sekcjach — algorytmy oraz metody i techniki informatyczne), **29 komunikatów** o nowych algorytmach oraz **dwie sesje panelowe** (dotyczące problemów tworzenia baz danych w systemach informatycznych oraz analizy, implementacji i rozwoju algorytmów). Wśród tematów podejmowanych na konferencji można wyodrębnić następujące grupy problemowe:

- nowe algorytmy numeryczne, zwłaszcza z zakresu algebry liniowej, metod kombinatorycznych, robotyki, rozpoznawania obrazów,
- metody projektowania, wdrażania i oceny efektywności systemów informatycznych
- metody organizacji i przechowywania danych w architekturze systemów informatycznych
- metody efektywnego dostępu do informacji przechowywanych w bazach danych.

Spośród 10 referatów programowych na szczególną uwagę zasługują takie tematy, jak:

- język rozpoznawania danych w systemie przetwarzania tekstów i zadań (M. Ciganik — Bratysława)
- prognozy rozwoju algorytmów (F. Klúska — Bratysława)
- algorytmy specjalizowanych macierzowych systemów komputerowych (J. Mikloško — Bratysława)
- efektywność systemów informatycznych (M. Bazewicz, J. Dębowy — Wrocław)
- ilościowe i jakościowe metody porównywania języków symulacyjnych (F. Stuchlik, P. Lorenz — Magneburg).

Słuszność wyboru problematyki oraz form organizacyjnych jej upowszechniania w Czechosłowacji potwierdza duża aktywność uczestników konferencji. Klimat zaangażowania towarzyszył zarówno posiedzeniom sesyjnym, jak i dyskusjom panelowym, w których oprócz pytań szczegółowych były wystąpienia polemizujące z tezami głoszonymi przez referentów, a także kontrowersyjne w stosunku do poglądów reprezentowanych przez panelistów i moderatorów posiedzeń panelowych (dr M. Ciganik, J. Mikloško).

Istotną zaletą tego rodzaju imprezy naukowej jest nie tylko publikowanie pełnych tekstów referatów, lecz systematyczne wydawanie specjalnego katalogu (zbioru) algorytmów[1], który jest udostępniany wszystkim uczestnikom kolejnych konferencji ALGORYTMY. Wszystkie prezentowane algorytmy zostały opisane w sposób ujednolicony, wg ogólnie przyjętych zasad i formatów, oraz zawierają pełną informację o ich właściwościach i przeznaczeniu. Są one zapisane w języku ALGOL 60, PASCAL, FORTRAN oraz w innych językach programowania, jak np. PL-I. W katalogu algorytmy zostały sklasyfikowane w 13 grupach tematycznych, z których każda dzieli się dalej na podgrupy. Ograniczę się do wyszczególnienia samych nazw tych grup:

- metody aproksymacji
- numeryczne różniczkowanie i całkowanie
- równania różniczkowe i całkowe
- funkcje specjalne
- rachunek macierzowy
- wartości własne i wektory własne macierzy
- układy równań liniowych
- równania matematyczne
- metody matematyczne w ekonomice, w tym liniowe i nieliniowe programowanie, dynamiczne i stochastyczne programowanie i inne
- probabilistyczne metody i procesy, w tym generowanie liczb przypadkowych, metody Monte Carlo, systemy masowej obsługi, metody symulacyjne i inne
- kombinatoryka
- inne metody.

Wydawanie katalogów algorytmów i upowszechnianie ich wśród uczestników konferencji sprzyja wzrostowi wykorzystania tych opracowań w różnych ośrodkach obliczeniowych, zacieśnianiu współpracy między różnymi zespołami autorskimi (np. konieczność aktualizacji algorytmów wynikających z obowiązku nałożonego na autorów) oraz szerokiej wymianie doświadczeń. O randze tak prowadzonej działalności może świadczyć fakt, że materiały kolejnych konferencji ALGORYTMY są gromadzone w zbiorach dokumentacyjnych informatyki w Moskwie i Paryżu. Wydaje się w pełni uzasadnione zamieszczenie na łamach INFORMATYKI wybranych, bardziej interesujących pozycji publikacyjnych konferencji ALGORYTMY. Nasze osiągnięcia są także prezentowane na tych spotkaniach. Szczególną aktywność wykazuje środowisko informatyków Wrocławia, Warszawy i Krakowa. Kompletne materiały z czterech ostatnich konferencji znajdują się w Politechnice Wrocławskiej i mogą być udostępnione na życzenie zainteresowanych.

Mieczysław BAZEWICZ

[1] Knižnica Algoritmov — VI diel, ALGORYTMY'81, Vysoke Tatry, Strbske Pleso 1981. Jednostka Slovenskych Matematikov a Fizykov pré SAV — OSVS Bratislava.

Pula papieru przeznaczona dla INFORMATYKI gwałtownie maleje. W poprzednim numerze zapowiedzieliśmy wydanie w tym roku dwóch numerów łączonych dzisiaj wiadomo, że będzie ich cztery tzn. wszystkie pozostałe. Prenumeratorom oddana zostanie nadwyżka przedpłaty. (Red.)

Nowy numeryczny układ scalony INTEL 8087

Numeryczne układy scalone w znacznym stopniu eliminują w mikroprocesorach rozbudowane oprogramowanie, jakie jest niezbędne do wykonywania obliczeń numerycznych. Ponadto umożliwiają one wielokrotne zwiększenie szybkości i dokładności działania systemu mikrokomputerowego. Nowy układ scalony INTEL 8087 może współpracować z mikrokomputerami zarówno 8-, jak i 16-bitowymi. Wcześniejszy tego typu układ, a mianowicie 9411 firmy ADVANCED MICRO DEVICES był przeznaczony dla systemów 8-bitowych.

INTEL 8087 ma duży repertuar operacji arytmetycznych, które wykonuje ok. 100 razy szybciej niż jest to możliwe w oparciu o programy arytmetyki zmiennoprzecinkowej realizowane w sposób tradycyjny. Jego szybkość i dokładność działania dorównują analogicznym parametrom minikomputerów i komputerów wyposażonych w układy elektroniczne arytmetyki zmiennoprzecinkowej.

Działając jako procesor współpracujący, nie zaś jako układ peryferyjny INTEL 8087 przetwarza taki sam strumień instrukcji jak typowy mikroprocesor. W momencie natrafienia na instrukcję (numeryczną) INTEL 8087 może natychmiast rozpocząć jej wykonywanie, natomiast mikroprocesor kontynuuje wykonywanie pozostałych instrukcji (nienumerycznych). Szybkość takiego współbieżnego układu wzrasta jeszcze bardziej, gdyż wewnątrz układu INTEL 8087 znajdują się dwa, równolegle działające procesory.

INTEL 8087 ma zbiór instrukcji obejmujący pełny zakres operacji arytmetycznych i przestępnych. Architektura wewnętrzna układu opiera się na strukturze 64-bitowej, natomiast obliczenia zmiennoprzecinkowe wykonywane są na liczbach 80-bitowych. Układ jest pełną implementacją proponowanego przez IEEE standardu dotyczącego arytmetyki zmiennoprzecinkowej. Procesor działa na kilku typach danych, a mianowicie na 16-, 32- i 64-bitowych liczbach całkowitych oraz 32- i 64-bitowych liczbach zmiennoprzecinkowych (pojedyncza i podwójna precyzja). Wszystkie typy danych są przekształcane na 64-bitowy format wewnętrzny układu, obliczenia zaś są wykonywane w maksymalnej precyzji, tzn. z wykorzystaniem 80 bitów (1 bit znaku, 15 bitów wykładnika i 64 bity ułamka). Format taki pozwala działać na liczbach z zakresu 10^{-4932} do 10^{4932} (całkowita liczba cząstek we wszechświecie oceniana jest na 10^{75}). Mimo że większość obliczeń praktycznych nie wymaga tak dużego zakresu liczb, to rozwiązanie takie zmniejsza prawdopodobieństwo przepełnienia w przypadku złożonych obliczeń. Dla takich przypadków jak przepełnienie, pierwiastek kwadratowy z liczby ujemnej i dzielenie przez zero w układach INTEL 8087 są wbudowane także procedury wykrywania błędów.

Oprócz podstawowych operacji arytmetycznych (dodawanie, odejmowanie, mnożenie, dzielenie oraz pierwiastek kwadratowy), zbiór instrukcji INTEL 8087 zawiera także operacje konwersji dwójkowodziesiętnej oraz obliczenia funkcji przestępnych (wykładnicze, logarytmiczne, trygonometryczne). Algorytm obliczania funkcji przestępnych umożliwia osiągnięcie poziomu błędów poniżej trzech jednostek na ostatnim miejscu liczby 80-bitowej (w reprezentacji maszynowej) przy szybkościach 100—200 µs.

Czasy działania układu INTEL 8087 są bardzo krótkie (rzędu 4 µs) w przypadku instrukcji logicznych i porównywania oraz 25 µs — w przypadku mnożenia w podwójnej precyzji. Wydaje się, że obecnie nie ma minikomputera lub komputera, który zapewniałby taki poziom dokładności przy podobnych szybkościach działania.

Ceną, którą trzeba było zapłacić za osiągnięcie takich parametrów, jest wielkość układu. Ma on stosunkowo duże wymiary, ponieważ długość każdej z jego krawędzi przekracza 280 mm. Nasunąć to może pytanie dotyczące wydajności i kosztu zastosowanych urządzeń produkcyjnych.

Oprac. K.M. na podstawie czasopisma HIGH TECHNOLOGY, kwiecień 1980

Nieporozumienie

Biblioteka Problemów — mimo przeszkód wydawniczych, z jakimi boryka się PWN — swymi 267 pozycjami popularnonaukowymi zyskała sobie trwałą sympatię czytelników. Niestety, kolejna pozycja tylko w części odpowiada dotychczasowemu standardowi [1].

Satyryk określiłby to dzieło, opublikowane w polskim przekładzie pod wielce obiecującym tytułem „Dzieje komputerów", jako *„dobre niemiecko-akademickie dzieło w złym tego słowa znaczeniu"*. Cóż bowiem składa się na owe „Dzieje"?

— Ha, mamy trochę archeologii, okraszonej napomknieniem o teorii Shannona oraz wspaniały traktat o inwazji filozofii arystotelesowskiej, przyprawiony nazwiskiem Gödela. Dalsza część to nauka historii muzułmańskiej, obrazowo szpikowana anegdotkami o Al-Chorezmim, z obowiązkową żonglerką terminem software, jako klamrą spinającą algorytmy Al-Chorezmiego z późniejszym (o tysiąc lat) elektronicznym przetwarzaniem danych. Przedostatni rozdział traktuje obowiązkowo o papieżu... cybernetyki (Wienerze) i jego sympatii dla patrona tej nauki (Leibniza); przy okazji napomyka się co nieco o konstruktorach maszyn liczących (mechanicznych). O s t a t n i rozdział natomiast poświęcony jest komputerom i cudom krzemowych miniatur...

Oryginalny tytuł dzieła brzmi: „Die Ahnen des Computers" (dosł. „Przodkowie komputerów"). Hans Kaufmann

[1]) Hans Kaufmann: Dzieje komputerów. PWN, Warszawa, 1980; Biblioteka Problemów; z oryginału niemieckiego (1974 r.) przełożył Marek Kęsy, 138 str.; 26 ilustracji; cena 26 zł.

opatrzył jednak swój, uroczy skądinąd lekkością przeskakiwania z tematu na temat, traktacik refleksyjny „amortyzującym" podtytułem: „Od pisma fenickiego do przetwarzania danych". W polskim tłumaczeniu książka powinna się nazywać „Zaranie komputerów" lub jakoś podobnie, wtedy budziłaby tylko niedosyt, a nie aż — oburzenie.

W „Dziejach komputerów" mówi się bowiem tylko o dwu komputerach (ENIAC o fantastycznej szybkości i EDVAC), dwóch językach (ALGOL, FORTRAN), dwu czcionkach komputerowych (CMC-7 i OCR-A), dwóch cybernetykach (Wiener, Brillouin), są fotografie dwóch informatyków (Zuse, von Neumann), wspomina się jeszcze o dwu dalszych (Turing, Shannon), bliżej omawia się tylko dwa pojęcia komputerowe (algorytm, pamięć). Wprawdzie rola układu dwójkowego w budowie współczesnych komputerów jest ogromna, ale ograniczanie się tylko do takich „binarnych" przykładów jest dowodem — łagodnie mówiąc — odwagi.

Gorzkie te słowa nie zmienią faktu, że przez niedokształconych humanistycznie fanatyków techniki książka Kaufmanna będzie czytana z wypiekami na twarzy, ba, może nawet pobudzi ich do twórczych skojarzeń. Oby tylko nie chcieli podobnej książki napisać o... dziejach lotów kosmicznych (recepta: chińskie smoki, Ikar, Dogonowie, Werner von Braun rodem z Wierzycy i lot balistyczny Shepparda; podbudowa filozoficzna może być skopiowana z „Dziejów Komputerów", z zacytowaniem Ptolemeusza i Kopernika, jako dowodem oczytania.

Dziejów komputerów nie nazwałbym książką szkodliwą. Ostatecznie nie została napisana przez informatyka. Nieporozumieniem jest po prostu jej tytuł. Osobiście książkę przeczytałem z ciekawością. Szczególnie zainteresowały mnie podobieństwa fonetyczne nazw drugich liter w alfabetach fenickim, synajskim, arabskim i greckim. Nauczyłem się sylabizować napis klinowy na Pałacu Dariusza. Rewelacją była też dla mnie uwaga o współczesnym zastępowaniu telekomunikacji przewodowej elektrycznej przez światłowodową, z erudycyjnym napomknięciem, że o posługiwaniu się światłem w łączności wspominał już mityczny Homer. Także nie wiedziałem, że Polibiusz — autor opisu interesującego kodu pochodniowego — został stracony w 47 r. n.e. jako sekretarz znanego nam z telewizyjnego serialu — Cesarza Klaudiusza. Uzupełniłem też swoją wiedzę geograficzną; zamieszczone mapki Jonii, krajów arabskich, a zwłaszcza Kalifatu Kordobańskiego pozwoliły mi wreszcie zapamiętać, że górny bieg rzeki Ebro pozostał na szczęście dla komputerów poza zasięgiem muzułmanów.

Dawno lektura książki popularnonaukowej nie wprawiła mnie w tak miły nastrój. Nie potrafiłem się jeszcze tylko nauczyć „szwabachy Gutenberga" ze słynnej Biblii (bo i taką reprodukcję zawierają „Dzieje Komputerów"). Przy tak olbrzymiej obfitości dywagacji, znakomicie podnoszących — jako bezprzedmiotowe w stosunku do komputerów — przyswajalność dzieła, już może zdziwić fakt, że na omówienie stosowanych w starożytności osobnych znaków na liczebniki zabrakło po prostu miejsca. Ale o tym, że najważniejszym wynalazkiem starożytności było wynalezienie osobnych symboli graficznych dla liczebników, czyli cyfr, na szczęście wiedziałem już z innych książek.

Homera! Chciałbym wiedzieć kto opiniował tę niefrasobliwą książeczkę? Jeżeli prywatne wydawnictwo ECON VERLAG GmbH drukuje zakalce intelektualne pod komputerową sukienką, to czy musi to robić i... PWN?

Adam B. EMPACHER

Aktualia sprzed lat

Na początku bieżącego roku ukazała się na półkach księgarskich książka Zofii Menet pt. „Mała informatyka. Środki i zastosowanie" [1]). Jest to kolejna pozycja wydana przez Państwowe Wydawnictwo Ekonomiczne w ramach popularnej serii „Informatyka w praktyce". Z redakcyjnego komentarza, umieszczonego na wewnętrznej stronie okładki można się dowiedzieć o celowości jej wydania. Wynika stąd, że małe i średnie przedsiębiorstwa nie mają możliwości zakupu elektronicznych maszyn cyfrowych, muszą mieć jednak odpowiednie środki techniczne do wspierania nowoczesnych systemów zarządzania... *„W zasięgu możliwości tych przedsiębiorstw znajdują się środki techniki przetwarzania danych, uzupełnione w razie potrzeby maszynami do liczenia* (? — przyp. mój). *Istnieje więc duża grupa maszyn o różnych możliwościach eksploatacyjnych. Określa się je mianem środków małej informatyki"*. Głównym celem Autorki było więc *„zwrócenie uwagi na niedoceniane czasem możliwości eksploatowanych maszyn tej grupy, wraz z zaprezentowaniem maszyn nowych, dotąd nie znanych i szerzej nie stosowanych"*.

Tak określony cel książki jest bardzo interesujący z punktu widzenia praktyki gospodarczej, tylko że — niestety — nie został zrealizowany. Autorka miała zaprezentować *„maszyny nowe, dotąd nie znane i szerzej nie stosowane"*, gdy tymczasem to, co przedstawia w części dotyczącej środków (str. 7—83), jest wręcz archaiczne. Maszyny przedstawione na stronach: 33, 37, 38, 40 — autorka opisywała już 13 lat temu (!) w swoim skrypcie [2]). Również 13 lat temu maszyny te dokładnie opisali A. Jarzembowski [3]) i T. Walczak [4]). Natomiast 12 lat temu maszyny małej i średniej mechanizacji, te same, które w 1980 r. opisuje Z. Menet, można było znaleźć w książce T. Peche [5]). Elektroniczne maszyny do fakturowania, do księgowania, automaty obrachunkowe i minikomputery, przedstawione w książce na str. 83—145, były z kolei w podobny sposób opisane przez H. Sobisa [6]) kilka lat temu, a także

[1]) Menet Z.: Mała informatyka. Środki i zastosowanie. PWE, Warszawa, 1980, wyd. I, nakł. 3000 + 230 egz., str. 170, 35 rys., cena 30 zł.

[2]) Menet Z.: Środki techniczne małej mechanizacji prac obliczeniowych. SGPiS, Warszawa, 1968

[3]) Jarzembowski A.: Organizacja zmechanizowanej rachunkowości. Mała i średnia mechanizacja prac ewidencyjno-obrachunkowych. WSE, Poznań, 1969, Zeszyt nr 82.

[4]) Walczak T.: Maszyny liczące. PWE, Warszawa, 1968.

[5]) Peche T.: Organizacja i mechanizacja rachunkowości. PWE, Warszawa, 1969.

[6]) Elementy organizacji i mechanizacji rachunkowości. Red. H. Sobis, WSE, Wrocław, 1971; Organizacja i automatyzacja rachunkowości. Red. H. Sobis, AE, Wrocław, 1976.

(i to lepiej) przez A. Nowakowskiego i W. Olejniczaka w ramach serii „Informatyka w praktyce"[7] — trzy lata temu. Wspomniana zapowiedź opisu nowości jest zatem nieporozumieniem.

Część książki poświęcona zastosowaniu środków małej informatyki również rozczarowuje. Występuje tu dziwne zjawisko: im wyższy stopień techniki przetwarzania tym mniej objętości poświęca Autorka na opis możliwości zastosowań; w przypadku maszyn małej techniki przetwarzania danych — 10 stron, maszyn do księgowania i fakturowania — 7 stron, automatów obrachunkowych i minikomputerów — 4 strony.

Zagadnienia programowania, omówione na kilku tylko stronach są siłą rzeczy tak skrócone, że ich wartość merytoryczna jest znacznie ograniczona.

Rozdział „Wybrane zagadnienia organizacji stosowania środków technicznych małej informatyki" w rzeczy samej stanowi encyklopedyczny opis projektowania systemów informatycznych i nie jest organicznie związany z przedstawionymi w książce maszynami.

Klasyfikacja maszyn liczących (a nie cyfrowych), przedstawiona na rys. 9, jest przestarzała — była opracowana i stosowana wyłącznie w dobie maszyn mechanicznych, nie zdaje natomiast egzaminu w dobie elektronicznej. Świadczą o tym chociażby kłopoty Autorki z zaliczeniem minikomputerów i automatów obrachunkowych jednoznacznie do konkretnej grupy maszyn liczących. Nie mogę się zgodzić również ze stwierdzeniem zawartym na str. 147: *„Minikomputery mogą mieć zastosowanie w przetwarzaniu scentralizowanym, występując jako urządzenia transmisji danych".* Jest to przesadne uproszczenie roli inteligentnych urządzeń końcowych, jaką mają spełniać minikomputery w zdalnych systemach informatycznych.

W okresie ostrego kryzysu wydawniczego, kiedy wartościowe pozycje czekają kilka lat w kolejce do drukarni, PWE nie może pozwalać sobie na luksus wydawania książek, których przydatność jest — delikatnie mówiąc — wątpliwa.

Jan STĘPNIEWSKI

[7] Nowakowski A., Olejniczak W.: Minikomputery biurowe. PWE, Warszawa, 1978.

Czy komputer może decydować?

W czechosłowackim miesięczniku „MECHANIZACE — AUTOMATIZACE — ADMINISTRATIVY" (nr 10/1980) ukazał się artykuł dr. Miroslava Šteca z Bratysławy pt. „Czy można automatyzować decyzje?" Autor zaczyna go słowami: *„Ze względu na to, że wielu czytelników nie wątpi o tym, iż można automatyzować decydowanie, a niektórzy nawet znają przypadki, że decydowanie już zostało zautomatyzowane — pytanie postawione w tytule tego artykułu byłoby zbyteczne, gdyby autor chciał na nie odpowiedzieć twierdząco".* W dalszym ciągu autor pisze: *„Jeżeli jednak będziemy zajmować się problematyką decydowania bardziej szczegółowo, to zauważymy, że odpowiedź na pytanie (...) bynajmniej nie jest łatwa".*

Autor przystępuje następnie do możliwie ścisłego zdefiniowania pojęć d e c y z j a i c z y n n o ś c i d e c y z y j n e. Powołuje się przy tym m.in. na pracę J. Zieleniewskiego „Teoria organizacji i zarządzania" (w tłumaczeniu czeskim). W pracy tej d e c y z j a jest określona jako „nieprzypadkowy wybór w toku czynności". Pozostali autorzy, na których powołuje się M. Štec, są także jednomyślni co do tego, że przy decydowaniu musi mieć miejsce wybór spośród kilku wariantów.

Dr Štec uważa, że za decyzję może być uważany tylko taki wybór, który jest dokonywany przez podmiot uprawomocniony (wyposażony w kompetencje) do decydowania. Nie jest natomiast decyzją wybór dokonywany na podstawie informacji według uprzednio, z góry ustanowionych kryteriów.

Komputera, będącego maszyną (a nie osobą fizyczną lub prawną), nie da się wyposażyć w jakiekolwiek kompetencje, bowiem program komputerowy i zawarty w nim algorytm przetwarzania danych, reprezentują jedynie sformalizowane decyzje oparte na założeniach programowych. Ex definitione wynika więc, że czynności decyzyjnych automatyzować nie można. Decyzje podejmuje przyszły użytkownik systemu informatycznego, wybierając (czyli — ustalając z góry) określone kryteria przetwarzania i przekazując je w ramach zamówienia komórce zajmującej się zaprojektowaniem i oprogramowaniem systemu. Decyzję podejmuje więc zawczasu użytkownik systemu, komputer zaś ją tylko w zaistniałych okolicznościach ujawnia.

Z innej strony podchodzi do tego zagadnienia dr Rainer Schwarz w artykule zamieszczonym w nr 10/1980 miesięcznika „RECHENTECHNIK — DATENVERARBEITUNG" wydawanego w NRD. Już sam tytuł tego artykułu „Instrumenty kierowania — jak wykorzystuje się matematykę, statystykę, cybernetykę i EPD do podejmowania decyzji" wskazuje na pogląd autora, iż ETO jest tylko jednym z narzędzi.

Proces decyzyjny jest — analogicznie jak wszystkie inne procesy pracy — połączeniem siły roboczej, przedmiotu pracy i środków pracy, a zatem wyróżnia się w nim: podmiot decyzji (kierownictwo), przedmiot decyzji oraz jej technikę (metody). Autor zaleca stosowanie ETO jako jedną z takich metod — m.in. do rozwiązywania modeli matematycznych, programowania heurystycznego, budowania systemów informacji dla kierownictwa. Zdaniem autora — stosowane metody (techniki) pracy kierownictwa, w tym również ETO, mają na celu nie tylko przetwarzanie danych, ale też dokonywanie uzgodnień i przygotowywanie rozstrzygnięć, niż zaś samo decydowanie.

A morał, jaki zdaniem recenzenta wypływa z obu artykułów jest ten, że na zleceniodawcy opracowania systemu informatycznego — bez względu na to czy jest to bezpośredni użytkownik, czy też jego jednostka nadrzędna — ciąży szczególny obowiązek i odpowiedzialność za wyraźne określenie: czym dysponuje, czego potrzebuje, czego chce i o co mu chodzi.

Stefan W. ZAWADZKI

Encyklopedia języków wysokiego poziomu

Nakładem WNT ukazała się pozycja bardzo potrzebna polskim użytkownikom komputerów [1]). Książka ta dotyczy proceduralnych języków programowania wysokiego poziomu. Jej autor, J. E. Nicholls zastrzega się jednak, że nie jest to podręcznik dla twórców języków. Książka powinna natomiast *„służyć programistom praktykom, którzy na co dzień używają języków wysokiego poziomu w różnych dziedzinach zastosowań"*. Podobnie charakteryzują tę pracę wydawcy polscy, których zdaniem książka *„może być także przydatna studentom kierunków informatycznych wyższych uczelni"*. Z tym stwierdzeniem trudno się jednak zgodzić w całej pełni.

Wydaje się, że dla profesjonalnych informatyków jest to publikacja zbyt ogólna. Brakuje w niej bardziej pogłębionej analizy źródeł i przyczyn wybranego rozwiązania każdego z przedstawionych tu problemów, takiej analizy, która pozwoliłaby na sformułowanie wskazówek metodycznych przydatnych przy tworzeniu języków wysokiego poziomu. Dlatego najważniejsze byłoby chyba stwierdzenie, że książka jest adresowana do tych użytkowników, którzy traktują komputer (wraz z jego oprogramowaniem podstawowym) jako sprawne narzędzie, pomocne przy realizacji zawodowych zadań. Pogłębienie przez nich znajomości budowy i zasad działania języków programowania, czyli — bezpośrednio używanego narzędzia, wydaje się ze wszech miar celowe, jest nawet niezbędne dla zrozumienia istoty programowania.

J. E. Nicholls stwierdza we Wstępie: *„(...) każdy kto bezpośrednio korzysta z komputera, korzysta z pewnej formy języka programowania"*. Stwierdza też, że: *„(...) język programowania jest łącznikiem między komputerami a użytkownikami. Dzięki językowi określany jest nowy komputer o własnościach odpowiadających potrzebom użytkownika"*. Ponadto, zdaniem autora: *„(...) zasady (projektowania języków) w równym stopniu odnoszą się do porjektowania programów użytkowych, a konstruowanie dowolnego dużego systemu ma wiele wspólnego z projektowaniem języka (...)"*. Kolejnym argumentem, przemawiającym za koniecznością wnikliwego studiowania struktury użytkowanych języków programowania, jest wreszcie cytowany pogląd B. Whorf'a, że *„(...) język determinuje nasze myślenie (...)"*.

J. E. Nicholls podzielił książkę na dwie części. Pierwsza — „Tło i podstawy teoretyczne" obejmuje pięć pierwszych rozdziałów. We Wstępie oprócz wyjaśnień, dlaczego twórcy oprogramowania użytkowego powinni studiować budowę języków, autor sformułował najważniejsze, jego zdaniem, właściwości języków wysokiego poziomu, a mianowicie:

- nie zmuszają one użytkownika do zajmowania się specyficznymi cechami komputerów
- dają możność przenoszenia programów, czyli zapewniają pewien stopień niezależności od komputera
- umożliwiają pisanie programów w sposób bardziej zwięzły, niż jest to możliwe w języku maszyny
- umożliwiają pisanie programów w terminologii właściwej do rozwiązywanych problemów.

Zastanawiając się nad wymienionymi „właściwościami", można dojść do wniosku, że dwa pierwsze są argumentami dość prymitywnymi, aczkolwiek jako cechy kryterialne mogą być skuteczne, natomiast dwa następne ujmują głębiej istotę problemu, ale jako kryteria wydają się z kolei zbyt ogólnikowe.

W drugim rozdziale — „Spojrzenie użytkownika" — autor podejmuje próbę określenia sposobu analizy poszczególnych języków, służącej ocenie ich przydatności dla użytkownika. Wymienia tu np. następujące cechy proceduralnych języków wysokiego poziomu, o których — jego zdaniem — należy pamiętać przy projektowaniu języka:

- łatwość użycia danego języka
- duża moc, wyrażana liczbą obliczeń wykonywanych przez program określonej wielkości
- możność przenoszenia oprogramowania
- łatwość poprawiania
- łatwość dokumentowania, np. samodokumentowanie.

Autor zwraca uwagę, że cechy te są spełniane przez obecne języki wysokiego poziomu tylko częściowo i w różnym stopniu. Wyjaśnia też, że powszechny styl pracy nad tworzeniem oprogramowania użytkowego: planowanie, rysowanie sieci działań, programowanie i testowanie, został dawniej ukształtowany w efekcie wzorowania się na takim stylu pracy, jaki narzuciło programowanie w języku maszynowym. Czytelnikowi brakuje w tym miejscu jakiegoś komentarza od autora, oceny, czy taki styl pracy jest obecnie równie dobry jak dawniej.

W tym samym rozdziale J. E. Nicholls sygnalizuje zagadnienia związane z poszukiwaniem nowych, skuteczniejszych — głównie w sensie niezawodnościowym — metod programowania. Celem tych poszukiwań jest *„zapewnienie możliwości konstruowania programów poprawnych, a nie znajdowanie coraz to lepszych metod poprawiania programów (skonstruowanych) z błędami (...)"*. Jedną z najbardziej znanych i uznawanych obecnie technik takiego skutecznego programowania jest rozpropagowane przez Dijkstrę i Wirhtha programowanie strukturalne.

W tym rodzaju programowania istotna jest możliwość początkowego pominięcia szczegółów i zaprojektowania jedynie ogólnego kształtu danego programu, by następnie — po sprawdzeniu — stopniowo go uszczegóławiać. Jest to tzw. konstruowanie programów przez kolejne przybliżenia. Autor zauważa tu, że do programowania strukturalnego nadają się języki o strukturze blokowej, a więc bardziej ALGOL czy PL/I, niż FORTRAN i COBOL, w których pojęcie bloku w ogóle nie istnieje. Do problemów i zasad programowania strukturalnego autor powraca jeszcze wielokrotnie.

Rozdział trzeci — „Zastosowania" — poświęcony jest analizie przydatności różnych języków. Zdaniem autora — potrzeby związane z różnymi zastosowaniami wpływały na projektowanie języków, powodując znaczne różnice między nimi. Najsilniej różnice te zaznaczyły się pomiędzy językami przeznaczonymi do zastosowań naukowych i administracyjnych. Wprawdzie niektóre z tych różnic wynikały z przyczyn natury historycznej, ale inne są skutkiem rzeczywistych rozbieżności, czy to w strukturach danych, czy w stosowanych metodach i technikach.

Autor stwierdza przy okazji, że wbrew obiegowym poglądom ALGOL i FORTRAN nie zawierają wszystkich narzędzi istotnych dla zastosowań naukowych. Zdaniem Nicholls'a wynika to z faktu, że języki te odzwierciedlają tradycyjny pogląd na analizę numeryczną. Natomiast stworzenie języka ogólnego przeznaczenia, jakim jest PL/I *„jest afirmacją (!) jedności i dowodzi, że łączenie cech potrzebnych różnym grupom zastosowań w jednym języku jest możliwe i pożyteczne"*. Warto tu jednak zauważyć, że czytelnik, nie prowadzący porównawczych studiów nad różnymi językami programowania, musi to ostatnie stwierdzenie przyjąć całkowicie na wiarę. Autor nie przeprowadza bowiem żadnego dowodu prawdy ani nie ilustruje swego stanowiska przykładami.

[1]) John E. Nicholls.: Struktura języków programowania. WNT, Warszawa, 1980, seria Informatyka, str. 516, nakł. 6000 + 250 egz, cena 120 zł.

W rozdziale czwartym — „Implementacje języków", po zdefiniowaniu różnic między kompilatorem a interpreterem, autor zwraca uwagę, że obie te techniki wpływają na kształt języków. Po podaniu krótkiego zarysu budowy kompilatora, Nicholls stwierdza, że wynikiem kompilacji jest nie tylko przetłumaczony, gotowy do wykonania program, ale również zbiór wydruków — komunikatów z procesu kompilacji. Te ostatnie stanowią podstawową informację przy poprawianiu programu. Powinno się więc ich prawidłową formę i zakres traktować jako część definicji języka. Autor podkreśla tu, że współcześnie takie podejście do kompilacji i kompilatorów nie jest popularne, formy wydruków w każdym języku są zupełnie dowolne. Powinno się — jego zdaniem — przyjąć ściślejsze, bardziej ujednolicone ich formy, co pozwoliłoby na stworzenie systematycznych metod diagnozowania błędów; powinno się traktować ten problem jako ważną i samoistną do pewnego stopnia, część wiedzy o oprogramowaniu.

Interesujący jest również pogląd, że: *„(...) ogromny postęp jakim jest programowanie w językach wysokiego poziomu wcale nie znalazł odbicia w konstruowaniu kompilatorów (...)"*. Niestety, i to stwierdzenie musi czytelnik przyjąć w pewnym sensie na wiarę, gdyż autor nie rozwinął swej myśli.

Rozdział piąty — „Podstawy teoretyczne" — poświęcony jest omówieniu pojęcia języka formalnego. Po scharakteryzowaniu samego pojęcia „system formalny" i sposobu, w jaki wyrasta ono z pojęć matematyki i logiki, autor omówił sposób analizy składni traktowanej jako system formalny, wyjaśnił też, w jaki sposób jest to zazwyczaj ujmowane w publikacjach. Przy lekturze tej partii książki czytelnik odnosi wrażenie, że autor w pewien sposób dystansuje się od takiego, zmatematyzowanego podejścia do problemów języka programowania. Jest to jednak tylko mgliste wrażenie. A szkoda, byłaby bowiem celowa bardziej wnikliwa analiza. Dość ogólne zaprezentowanie tych trudnych problemów, pozostawia nie znającego teorii czytelnika w sytuacji „oniemiałego profana".

Przy lekturze pierwszej części książki czytelnik ma często trudności ze śledzeniem myśli autora, odnosi wrażenie licznych niedomówień i niejasności. Nie występuje to przy lekturze drugiej, bardziej przejrzystej części, która obejmuje siedem następnych rozdziałów i nosi ogólny tytuł „Elementy proceduralnych języków programowania". Jest to systematyczna analiza praktycznych aspektów konstruowania języków programowania, wskazująca na sposoby studiowania i analizowania tych języków.

Rozdział szósty omawia precyzyjnie poszczególne elementy języków, jak np. znaki, słowa kluczowe, wyrażenia, zdania, bloki, procedury, podprogramy. Po ogólnym scharakteryzowaniu tych elementów J. E. Nicholls pokazuje jak egzemplifikują się one w strukturach poszczególnych języków. W podobny sposób traktuje autor wszystkie zagadnienia omawiane w dalszych rozdziałach, a więc: dane, zmienne i ich deklaracje (rozdział siódmy) oraz struktury danych (rozdział ósmy), wyrażenia i sposoby przypisywania (rozdział dziewiąty).

W rozdziale dziesiątym omówiona jest kolejność wykonywania i sposoby sterowania kolejnością. W tym właśnie rozdziale szczególnie dużo uwagi poświęcał autor na analizę konsekwencji, jakie wynikają z wprowadzania zasad programowania strukturalnego, w aspekcie znaczenia poszczególnych instrukcji sterujących. W rozdziale jedenastym zaprezentowane zostały podstawowe wiadomości o wejściu i wyjściu, a w dwunastym — o budowie oraz znaczeniu podprogramów i procedur, o zasadach ich tworzenia, działania i stosowania.

Cały ten przegląd obejmuje cztery najważniejsze — zdaniem autora — języki wysokiego poziomu: ALGOL 60, FORTRAN, COBOL i PL/I. W paru miejscach autor rozszerzył swoje rozważania o analizę innych języków: APL czy LISP; sięgnął też do przykładów języków zupełnie nieznanych przeciętnemu użytkownikowi komputerów. Partie te są na ogół bardzo ciekawe, pokazują bogactwo możliwości przy projektowaniu języków specjalizowanych. Systematyczna, krótka charakterystyka czterech podstawowych języków zawarta jest w Dodatku.

Wydanie „Struktury języków programowania" umożliwia przeciętnemu polskiemu użytkownikowi komputerów dostęp do solidnej, encyklopedycznie wyłożonej wiedzy o zasadach budowy proceduralnych języków programowania wysokiego poziomu oraz do niezmiernie bogatego wykazu literatury tematu, co już samo w sobie może przynieść wiele pożytku.

Stanisława BONKOWICZ-SITTAUER

O JEDNOLITĄ TERMINOLOGIĘ

Teleinformatyka

Kontynuując rozważania na temat terminologii z pogranicza informatyki i telekomunikacji, warto zwrócić uwagę na termin m u l t i p l e k s e r, występujący w obu dziedzinach. Niedawno skrytykowano sposób jego użycia, przypominając, że odpowiednie urządzenie jest w telekomunikacji od wielu lat nazywane k r o t n i c ą (W. Nowicki, Przegląd Telekomunikacyjny, nr 8/1979). Spróbujemy wyjaśnić, w jakim znaczeniu termin ten występuje w informatyce.

Według projektu normy ISO DIS 2382/IX angielski wyraz m u l t i p l e x e r w transmisji danych oznacza jednostkę funkcjonalną (sprzętową lub programową — przyp. J. Z.) umożliwiającą dzielenie wspólnego ośrodka transmisji przez więcej źródeł danych niż wynika z liczby kanałów dostępnych w tym ośrodku. Natomiast sam podział ośrodka transmisji (toru — przyp. J. Z.) na dwa lub więcej kanałów określa się słowem *multiplexing* (Publ. IEC 65A/Sec/25).

Przypomnijmy (W. Nowicki, op. cit.), że czynność podziału jest w telekomunikacji od dawna określana po polsku jako u w i e l o k r o t n i e n i e t o r u. Jednakże w informatyce podział oznacza nie tylko zasadę wykorzystania toru, lecz także krótkotrwałą czynność zmiany kanałów podczas transmisji. To drugie znaczenie terminu *multiplexing* nie jest, niestety, ujęte w normach; choć jest oczywiste.

Czynność tę najłatwiej zilustrować działaniem urządzenia pracującego w systemie gromadzenia danych, które do pojedynczego przetwornika analogowo-cyfrowego doprowadza sygnały analogowe z wielu źródeł. Kolejne dołączanie źródeł A, B, C itd. do przetwornika określane jest terminem *multiplexing.* W systemie mikroprocesorowym terminem tym określa się zaś naprzemienne przesyłanie danych i adresów przez magistralę w jednym cyklu.

W powyższych przykładach słowo u w i e l o k r o t n i e n i e nie odpowiadałoby znaczeniu, w jakim użyto wyrazu *multiplexing*.

W różnych pracach próbuje się nazwać tę czynność p r z e ł ą c z a n i e m sygnałów lub danych. Zwróćmy jednak uwagę, że w języku polskim czasownik p r z e ł ą c z a ć odnosi się na ogół tylko do przedmiotów np. przełączyć element X z położenia Y na Z (W. Nowicki, Przegląd Telekomunikacyjny, nr 12/1979) lub przełączyć telefon (Słownik poprawnej polszczyzny, PWN, Warszawa, 1976). Gdyby należało trzymać się tej reguły to trzeba by używać sformułowania — p r z e ł ą c z a n i e ź r ó d e ł s y g n a ł ó w, a nie — p r z e ł ą c z a n i e s y g n a ł ó w. Należy jednak przyznać, że złamanie tej zasady, byłoby znacznie łagodniejszym zabiegiem dla języka niż wprowadzenie czasownika m u l t i p l e k s o w a ć, jak postępuje wielu autorów. Natomiast, biorąc pod uwagę powyższe różnice znaczeń, głosowałbym za usankcjonowaniem powszechnego używania terminu m u l t i p l e k s e r jako odpowiednika angielskiego *multiplexer*.

W elektronice stosunkowo często multiplekser bywa nazywany k o m u t a t o r e m. W informatyce jednak mogłoby to prowadzić do nieporozumień, choćby dlatego, że termin k o m u t a c j a (ang. *switching*) ma znaczenie ustalone, przejęte z telekomunikacji i służy do określania czynności związanych z ustalaniem połączeń między stacjami.

Dla informacji podajemy poniżej odpowiednie definicje zawarte w wymienionej propozycji normy ISO.

K o m u t a c j a ł ą c z y (ang. *circuit switching)* jest to czynność polegająca na połączeniu (na żądanie) dwóch lub większej liczby urządzeń końcowych i umożliwieniu im wyłącznego użycia łącza aż do przerwania połączenia.

K o m u t a c j a w i a d o m o ś c i (ang. *message switching)* jest to czynność polegająca na wyborze trasy wiadomości w sieci transmisji danych przez odbieranie, zapamiętywanie i wysyłanie pełnych wiadomości.

K o m u t a c j a p a k i e t ó w (ang. *packet switching)* jest to czynność polegająca na wyborze trasy i przekazywaniu danych za pomocą zaadresowanych pakietów w taki sposób, aby kanał był zajęty jedynie podczas transmisji pakietu, a po jej zakończeniu — udostępniony do transmisji pakietu następnego.

Według tego samego projektu w i a d o m o ś c i (ang. *message)* jest to grupa znaków (w porządku określonym przez źródło) i bitów kontrolnych przesyłanych jako całość ze źródła do ujścia danych. Natomiast przez p a k i e t (ang. *packet)* rozumie się dane i ciąg bitów kontrolnych w odpowiednim formacie, przesyłane w całości określonej przez proces transmisji.

Na zakończenie wypada stwierdzić, że przedstawienie niektórych terminów z zakresu teleinformatyki; zawartych w propozycji ISO DIS 2382/IX, czego dokonaliśmy w numerach 4—7 INFORMATYKI, nie powinno przesądzać ich ostatecznego przyjęcia. Ostatnie słowo musi należeć do specjalistów. Przede wszystkim należałoby zbadać zgodność podanych określeń z polskimi normami terminologicznymi, w szczególności chodzi o nazwy dróg przesyłowych sygnałów (por. PN/T-01001 — Słownictwo telekomunikacyjne. Pojęcia podstawowe). Warto również zastanowić się, na ile określenia te mogą odbiegać od dotychczas używanych w literaturze polskojęzycznej.

Janusz ZALEWSKI

KONFERENCJE

SPIS'81

Założenia reformy systemu planowania i zarządzania gospodarką narodową przewidują daleko idące zmiany w funkcjach, organizacji i zasadach działania organów i instytucji obsługiwanych przez centralne systemy informatyczne (CSI). Wypracowany na początku lat siedemdziesiątych model tych systemów, wyróżniający systemy rządowe i resortowe na szczeblu centralnym oraz systemy informatyczne zjednoczeń i regionów, nastawiony był na obsługę informacyjną hierarchicznego, scentralizowanego systemu podejmowania decyzji, zaniedbując obsługę funkcji analitycznych i koordynacyjnych tych jednostek. Wskutek tego, zwłaszcza w systemach resortowych i zjednoczeniowych skoncentrowano się na gromadzeniu i opracowywaniu informacji niezbędnych do operatywnego kierowania jednostkami podległymi. W tym kierunku rozwijano systemy informowania kierownictwa i banki danych oraz kształtowano organizację gromadzenia informacji.

Reforma gospodarcza przewiduje zasadniczą zmianę proporcji w układach pionowych struktur organizacyjnych między funkcjami analityczno-koordynacyjnymi a funkcjami operatywnego zarządzania na korzyść tych pierwszych.

Istnieje więc potrzeba „samookreślenia" funkcji i rozwiązań systemowych CSI w dostosowaniu do zmian modelu zarządzania. Chodzi o to, aby na podstawie istniejącego rozpoznania kierunków zmian systemu planowania i zarządzania określić:

- **zadania i funkcje CSI w warunkach zreformowanej gospodarki, uwzględniając zarówno nowe potrzeby informacyjne centralnej i regionalnej administracji gospodarczej, jednostek koordynacji branżowej (poziomej), a także organów władzy państwowej, związków zawodowych oraz innych instytucji społecznych**
- **możliwości podjęcia przez CSI działań zapewniających ciągłość obsługi informacyjnej w procesie wdrażania reformy; chodzi o to, aby wskazać, na ile i w jaki sposób CSI mogą ułatwiać wprowadzenie nowych rozwiązań systemu zarządzania gospodarką, na ile zaś petryfikacja dotychczasowych rozwiązań funkcjonalno-organizacyjnych stanowić może czynnik utrudniający organom planowania i zarządzania podejmowanie nowych funkcji**
- **strategię przejścia od dotychczasowego modelu CSI do modelu docelowego, w sposób gwarantujący wykorzystanie dotychczasowego dorobku tych systemów i ich potencjału metodycznego, kadrowego i technicznego**

Konfrontacja poglądów oraz przedyskutowanie koncepcji rozwiązań systemowych i funkcjonalno-organizacyjnych jest głównym celem seminarium SPIS'81. Konfrontacja powinna umożliwić wypracowanie wspólnego stanowiska środowiska informatyków i użytkowników CSI w trzech wymienionych wyżej obszarach problemowych, a przynajmniej wzajemne zrozumienie celów i problemów wyznaczających miejsce i zadania różnych typów CSI wobec założeń reformy gospodarczej.

Obrady seminarium koncentrować się będą na następujących obszarach problemowych:

- **zmiany i próba określenia potrzeb informacyjnych centralnych i regionalnych organów planowania i zarządzania w świetle założeń reformy gospodarczej**
- **bariery informacyjne wdrażania reformy gospodarczej i możliwości ich przezwyciężania z pomocą CSI**
- **docelowe modele CSI w warunkach zreformowanej gospodarki — cele, zadania i rozwiązania funkcjonalno-organizacyjne systemów ogólnopaństwowych (rządowych), resortowych, branżowych i regionalnych**
- **problemy i strategia przekształcenia obecnie ukształtowanych systemów informatycznych w procesie wdrażania reformy gospodarczej (od stanu istniejącego do modeli docelowych).**

Seminarium przeznaczone jest dla naukowców, analityków i projektantów, statystyków oraz użytkowników zainteresowanych problematyką CSI.

Obrady odbywać się będą w Ośrodku Doskonalenia Kadr Kierowniczych i Szkolenia Zawodowego Głównego Urzędu Statystycznego w Jachrance w pierwszej dekadzie grudnia 1981 r.

Bibliografia wydawnictw polskich z dziedziny informatyki

● **Projektowanie w języku BASIC — ISZKOWSKI W., MANIECKI M. PWN, Warszawa 1980, s. 247, cena 34 zł**

Podstawy programowania w języku BASIC. Rozszerzenia języka BASIC. Uruchamianie programów. Dodatki.
Materiały są przeznaczone dla studentów kierunków informatycznych oraz programistów.

● **Simula 67 — OKTABA H., RATAJCZAK W., WNT, Warszawa 1980, s. 195, cena 60 zł**

Klasy obiektów. Prefiksowanie klas i typy referencyjne. Dostęp do atrybutów obiektów. Blok prefiksowany. Klasa systemowa SIMSET. Przekazywanie parametrów. Systemy quasi-równoległe. Programowanie symulacji. Klasy SIMULATION. Znaki, teksty, we/wy. Dodatki: Składnia. Treść klasy SIMULTATION. Procedury pseudolosowe.
Materiały są przeznaczone dla programistów oraz dla studentów kierunków informatycznych.

● **Fortran dla maszyn ODRA serii 1300 — PACHELSKI W., WNT, Warszawa 1980, s. 241, cena 75 zł**

Podstawowe struktury językowe. Stałe zmienne, tablice i wywoływania funkcji. Wyrażenia arytmetyczne i logiczne oraz instrukcje. Instrukcje sterujące, we/wy oraz dyrektywy specyfikacji zbiorów i rekordów zewnętrznych. Funkcje i podprogramy. Dyrektywy COMMON i EQUIVALENCE. Inicjalizacja zmiennych i tablic. Kompilowanie i wykonywanie programów na maszynach ODRA serii 1300. Dodatki.
Książka jest przeznaczona przede wszystkim dla pracowników naukowych i inżynierów, którzy w swojej pracy zawodowej posługują się komputerem. Może być również przydatna programistom i studentom kierunków informatycznych.

● **Mikroprocesory i minikomputery — BELICZYŃSKI B. i inni, Wyd. Politechniki Warszawskiej, Warszawa 1980, s. 166, cena 14 zł**

Organizacja mikrokomputerów. Oprogramowanie. Problemy łączenia mikroprocesorów z otoczeniem. Dodatki.
Skrypt jest przeznaczony dla studentów sekcji Automatyzacji Procesów Technologicznych, Energoelektroniki i Miernictwa Wydziału Elektrycznego.

● **Sterowanie optymalne oraz układy mikrokomputerowe — SWINIARSKI R. Wyd. Politechniki Warszawskiej, Warszawa 1980, s. 377.**

Optymalizacja dynamiczna. Pewne numeryczne metody wyznaczania sterowań czaso-optymalnych liniowych układów stacjonarnych. Algorytmy realizacji strategii sterowania czaso-optymalnego w czasie rzeczywistym. Rozwiązania, algorytmy, programy sterowań czaso-optymalnych dla liniowych i nieliniowych procesów cieplnych. Algorytmy i programy dla sterowań optymalnych ze względu na wydatek. Minikomputerowe i mikrokomputerowe stuktury sterowania optymalnego w czasie rzeczywistym. Sterowanie suboptymalne. System programów do wyznaczania sterowań czaso-optymalnych, realizacji i symulacji sterowań optymalnych w czasie rzeczywistym, rozwiązań i symulacji stacjonarnych liniowych układów sterowania. Pewne algorytmy i programy dla optymalnych i suboptymalnych liniowych wskaźnikach jakości. Problemy wrażliwości. Algorytmy dla mikrokomputerów. Mikrokomputerowe systemy sterowania sekwencyjnego.
Materiały są przeznaczone dla studentów Wydziału Elektrotechniki o specjalizacji elektroniki przemysłowej, mogą być również przydatne dla inżynierów automatyków.

● **Architektura komputerów — MANOM M., tłum. wyd. ang. z 1976 r., WNT, Warszawa 1980, s. 450, cena 110 zł**

Cyfrowe układy logiczne. Układy scalone i funkcje cyfrowe. Przedstawienie danych. Przesłania międzyrejestrowe i mikrooperacje. Organizacja i projektowanie prostego komputera, jego programowanie. Organizacja jednostki centralnej. Sterowanie mikroprogramowe. Projektowanie arytmometru. Algorytmy arytmetyczne. Organizacja we/wy. Organizacja pamięci.
Książka przeznaczona jest w szczególności dla studentów wydziałów informatyki i elektroniki. Może być przydatna projektantom i użytkownikom systemów komputerowych.

● **Algorytmy + struktury danych = programy — WIRTH N., tłum. wyd. niem. z 1976 r., WNT, Warszawa 1980, s. 376, cena 69 zł**

Podstawowe struktury danych. Sortowanie. Algorytmy rekurencyjne. Dynamiczne struktury informacyjne. Struktury językowe i kompilatory. Dodatki: Zbiór znaków ASCII. Diagramy składni dla PASCAL-a.
Książka przeznaczona jest dla programistów.

● **Architektura minikomputerów — KOMOROWSKI W., WNT, Warszawa 1980, s. 255, cena 42 zł**

Struktura sprzętowa komputera. Kodowanie informacji. Arytmometr. Pamięć operacyjna. Działanie procesora centralnego. Budowa rozkazu. Repertuar rozkazów. Struktura logiczna procesora. Przerwania. Układ sterowania. Sterowanie mikroprogramowe. Współdziałanie urządzeń zewnętrznych. Organizacja pamięci komputera. Mikroprocesory. Modułowe systemy pomiarowe i sterujące. Architektura minikomputerów PDP-11.
Dodatek: Zasady formalnego opisu architektury komputerów.
Książka zawiera wprowadzające wiadomości z zakresu organizacji i projektowania minikomputerów. Przeznaczona jest dla inżynierów elektroników i informatyków oraz studentów wyższych szkół technicznych.

● **Grafoskopy w systemach komputerowych — Praca zbiorowa, WNT, Warszawa 1980, s. 327, cena 63 zł**

Budowa i działanie grafoskopów, ich oprogramowanie i zastosowanie. Książka przeznaczona jest dla informatyków, projektantów i użytkowników systemów komputerowych.

● **Komputery w kierowaniu pracą systemu elektroenergetycznego — GŁADYS H., WNT, Warszawa 1980, s. 266, cena 88 zł**

Charakterystyki eksploatacyjne systemu elektroenergetycznego i jego elementów. Problemy dyspozytorskiego kierowania pracą tego systemu. Konfiguracja systemów komputerowych i ich oprogramowanie podstawowe. Metody i algorytmy obliczeniowe. Systemy informatyczne do planowania pracy systemu elektroenergetycznego (tryb autonomiczny). Systemy informatyczne do sterowania i kontroli pracy systemu elektroenergetycznego (tryb nadążny).
Książka przeznaczona jest dla inżynierów energetyków, w szczególności zaś dla pracowników dyspozycji mocy i ruchu.

● **Lisp. Opis, realizacja i zastosowania — MARTINEK J., WNT, Warszawa 1980, s. 203, cena 64 zł**

Struktura danych czystego Lispu. Metajęzyk i zapis listowy funkcji. Semantyka. Struktury danych i sterowania Lispu 1.5. Wprowadzanie i wyprowadzanie danych. Interpretator Lispu 1.5. Konstruowanie funkcji. Przykłady zastosowań. Kompilator Lispu. Szczegóły realizacji systemu.
Książka przeznaczona jest dla programistów oraz projektantów systemów informatycznych.

● **Minikalkulatory w obliczeniach naukowych i technicznych — KOZARSKI M., SZURMAK Z., WNT, Warszawa 1980, s. 447, cena 95 zł**

Elektroniczne minikalkulatory. Metody numeryczne. Wybrane problemy zastosowań naukowych i technicznych. Dodatki.
Książka przeznaczona jest dla szerokiego kręgu czytelników — użytkowników kalkulatorów, w szczególności dla pracowników biur projektowych i instytutów naukowych oraz studentów wyższych szkół technicznych.

● **Niezawodność oprogramowania — KOPETZ H., tłum. wyd. niem. z 1976 r., WNT, Warszawa 1980, s. 135, cena 42 zł**

Pojęcia podstawowe. Błędy. Struktura oprogramowania. Wymagania funkcjonalne. Uwzględnienie niezawodności w projekcie systemu. Weryfikacja oprogramowania. Ręczne usuwanie błędów. Automatyczne wykrywanie błędów i ich usuwanie. Konserwacja oprogramowania. Zarządzanie realizacją niezawodnego oprogramowania.
Książka przeznaczona jest dla programistów, analityków systemów oraz studentów kierunków informatycznych wyższych uczelni.

Opr. A.K.

Do Redakcji

Informatyka to nauka dotycząca wszelkiej maszynowej obróbki informacji. Powołuję się na tę lakoniczną definicję, aby zmusić Was moralnie do zajęcia się także problemami informacji. Jak dotąd, w większości przypadków wszelkie przetwarzanie informacji to nic innego jak przerzucanie pustego w próżne. Wynika to przede wszystkim z małej wartości informacji użytej na wejściu maszyny cyfrowej (np. symbole cyfrowe, czyli tzw. kody). Chciałbym abyście trochę więcej miejsca na łamach pisma przeznaczyli na kody, ale nie maszyn cyfrowych — tylko informacyjne, np. Kody materiałowe.

Historycznie ujmując, żadna — jak dotąd — siła w Polsce nie zmusiła zakładów pracy do stosowania jednolitego kodu materiałowego. A przecież najlepsza nawet maszyna cyfrowa czy system informatyczny są bezwartościowe, jeżeli użyty kod jest mało wydajny. I choć z komputerów i systemów robicie ołtarz, a z informatyków kapłanów, to pamiętajcie, że najważniejsi są wierni, czyli użytkownicy informacji wyjściowej.

Żaden kościół nie utrzyma się, jeżeli wiara głoszona przez niego będzie nie do przyjęcia. A tak jest z informatyką, która umrze śmiercią naturalną z chwilą przejścia zakładów pracy na własny rozrachunek. Cała dotychczasowa informatyka trzyma się tylko na podstawie nakazów z góry, wspieranych dotacjami państwowymi. A te się w najbliższym czasie skończą. Chcąc uniknąć śmierci trzeba poprzez Wasze czasopismo zmusić informatyków do zmiany dotychczasowego sposobu myślenia: najważniejsza jest informacja, która podlega przetwarzaniu, a reszta jest tylko dodatkiem.

Jeżeli ktokolwiek z Was to zrozumie, to proszę zwrócić się do mnie, a rozpoczniemy akcję modyfikacji informacji, polegającą na ujednolicaniu — w skali PRL — wszelkich kodów, począwszy od materiałowego, a skończywszy na symbolach Polskich Norm.

Adam **JAGŁA**
Poznań

OD REDAKCJI

Powyższy list odzwierciedla aktualne, niestety coraz bardziej krytyczne poglądy „szeregowego" użytkownika na rzeczywistą efektywność wielu eksploatowanych w kraju systemów informatycznych. Mimo nieco uproszczonej argumentacji Autora listu, projektanci i realizatorzy sytemów powinni traktować wypowiedź tę jako istotny głos szerokiej opinii społecznej.

Szanowny Panie Redaktorze

Zwracam się do Pana z nieprzyjemną i niezręczną sprawą. W INFORMATYCE (nr 11/1980) ukazał się artykuł L. Dobruckiego pt. „Wybór systemu zarządzania bazą danych dla INFOCHEM-u". Artykuł ten składa się z fragmentów większego opracowania L. Dobruckiego i M. Leśnego, przygotowanego w ZETO Gdynia, na zlecenie Gdańskich Zakładów Nawozów Fosforowych.

Publikowaną w INFORMATYCE pracę L. Dobrucki przedstawił wcześniej na V kursokonferencji z cyklu „Informatyka" (22—23 kwietnia ub.r.) organizowanej m.in. przez TNOiK w Bydgoszczy. Zgłaszając referat na konferencję, przedstawił się jej organizatorom jako jedyny jego autor. W wyniku interwencji osób znających wcześniejsze opracowanie, stwierdził jednak w trakcie wygłaszania referatu, że zostałem pomyłkowo pominięty jako współautor.

Mimo tych doświadczeń, publikując artykuł w INFORMATYCE, L. Dobrucki znowu pominął mój udział w przygotowaniu tego opracowania. Nie zawiadomił mnie także o zamiarze publikacji, czego bardzo żałuję, ponieważ zaproponowałbym uaktualnienie artykułu. Warto byłoby — na przykład — prześledzić jak zmieniły się możliwości badanych wówczas (ww. opracowanie powstało w 1977 roku) SZBD — np. parametry techniczne, kalkulacja kosztów — ceny zakupu, opłaty dzierżawne, itp. Z przykrością informując Pana o tym incydencie — łączę wyrazy szacunku.

Maciej **LEŚNY**
CPIZI, Warszawa

OD REDAKCJI

Powyższy list przesłaliśmy p. Leszkowi Dobruckiemu z prośbą o wyjaśnienie sprawy. Odpowiedzi — mimo zapewnień — nie otrzymaliśmy. Ostatnio okazało się ponadto, że p. Dobrucki wyjechał za granicę. Postanowiliśmy zatem dłużej na odpowiedź nie czekać.

Dokończenie z III strony okładki

celów: mikroprocesory mogą eksportować tylko nieliczne firmy światowe, natomiast wielu urządzeń bez wbudowania sterowania mikroprocesorowego eksportować dłużej niesposób...

Nie chciałabym urazić p. inż. Jacka Karpińskiego, poddając pod dyskusję Jego pomysł kolejnego zmierzenia się z czołówką światową (mam bowiem dla Niego ogromny szacunek i uznanie).

Pytam jednak wprost: a co ma Pan dla rodaków, Panie Inżynierze?

Zwracam się więc do p. inż. Jacka Karpińskiego z uprzejmą prośbą, aby zechciał osobiście zabrać — oby jak najszybciej — głos na tych łamach, przedstawiając swoje koncepcje i zamiary. Czekamy na Pana! W szczególności proponuję poruszenie następujących zagadnień:

— czy uzasadniona jest w obecnej sytuacji gospodarczej produkcja dużej ilości sprzętu komputerowego dla użytkowników krajowych?

— jeżeli tak, to czy jest możliwe osiągnięcie sensownego kompromisu między maksymalizacją osiągów technicznych a minimalizacją importu podzespołów?

— jeżeli tak, to jak przedstawiałaby się koncepcja sprzętu do takiego systemu?

— jakie byłoby początkowe minimum oprogramowania?

— jakie warunki powinny być spełnione, aby przy sensownych nakładach wstępnych, można było uruchomić szybko dużą produkcję, zachowując umiarkowaną cenę wyrobu i jego odpowiednią jakość?

Sądzę, że tysiące „szarych zjadaczy informatycznego chleba" z wielką uwagą przeczytałoby wypowiedź p. inż. J. Karpińskiego właśnie w INFORMATYCE.

Łódź, 5 kwietnia 1981 r.

Krystyna **ŻEBROWSKA**

KRYSTYNA ŻEBROWSKA

Odnowa INFORMATYKI?

Od kilku miesięcy powtarzają się apele Redakcji: „piszcie do nas", „poszukujemy współpracowników", „łamy otwarte dla wszystkich". Pomimo licznych wątpliwości, zdecydowałam się jednak napisać. Niemniej rodzi się pytanie, czy rzeczywiście nasze jedyne czasopismo fachowe chce zmienić swą formę i treść, aby w duchu odnowy oddziaływać na środowisko? Doświadczenia wynikające z ostatnich wydarzeń społecznych mówią bowiem, że proces odnowy często jest pozorny i że czeka nas niejedna ciężka próba sił.

Sądząc po zawartości ostatnich numerów INFORMATYKI (do nru 2/81 włącznie) — liczne zachęty Redakcji do nawiązania kontaktów pozostają bez echa. Można więc wnioskować, że mimo wszystko Czytelnicy bardzo mało piszą. Nie wierzę w to, że nie czują się na siłach czy nie mają czasu. Jest o czym pisać i wiem, że wielu informatyków ma własne, oryginalne poglądy na sprawy żywo interesujące nas wszystkich.

Ludzie nie piszą, ponieważ — jak sądzę — n i e w i e r z ą:

— w przyszłość tego wspaniałego zawodu, któremu w naszym kraju zmarnowano wiele możliwości

— że istnieją jeszcze osoby, które myślą nie tylko o swoich karierach, dochodach lub zakupach w godzinach pracy

— że może być prawdziwa dyskusja, w której zostaną uważnie wysłuchani, a ich rzeczowe argumenty — wzięte pod uwagę

— że winni istniejącego stanu rzeczy będą ocenieni i rozliczeni, a nowi decydenci zrozumieją, że mają służyć krajowi i ludziom

— że Redakcja będzie również walczyła o rangę czasopisma oraz o to, aby głos środowiska wyrażony na jego łamach liczył się na szerszym forum.

Nie możemy strajkiem popierać naszych racji. A sama często wątpię, że wystarczy nam sił na pogłębienie i utrwalenie odnowy informatyki. Czego zaś oczekuję teraz od Redakcji? Po pierwsze — programu minimum ratowania informatyki. Program ten może być skromny, ale realny i przekonywający. Powinien on dawać szansę na osiągnięcie szybkiego, odczuwalnego efektu. Taki program powinna jak najszybciej przedstawić nieformalna grupa osób skupiona wokół Redakcji, która notabene — może tu odegrać wielką rolę. Po drugie — zapewnienia, że żadne pytanie postawione przez Czytelników nie pozostanie bez odpowiedzi. Wprawdzie zdaję sobie sprawę, że większość instytucji i osób nie jest przyzwyczajona do publicznego rozliczania się ze swej działalności, bez tego jednak nie może być mowy o oczyszczeniu atmosfery w branży. Po trzecie — jasnego określenia roli naszego miesięcznika w zmienionej sytuacji. Nowe zadania INFORMATYKI oraz jej zaangażowanie się w proces odnowy wymagają — jak sądzę — daleko idących zmian w stylu pracy autorów i Redakcji, wreszcie — w profilu całego czasopisma i kształcie każdego numeru. Krótko mówiąc — INFORMATYKA ma być dla Czytelników, a nie dla autorów.

Uważam, że jedną z najważniejszych spraw, które w najbliższym czasie powinny być wszechstronnie i wyczerpująco omówione w INFORMATYCE, jest sprawa sprzętu. Bez niego działalność zawodowa informatyków nie ma racji bytu. Moim zdaniem, niedobrze się stało, że zagadnienia sprzętu komputerowego zajmowały na łamach INFORMATYKI trzeciorzędne miejsce.

31 marca br. ukazała się w prasie codziennej informacja, że ministrowie nauki, szkolnictwa wyższego i techniki oraz przemysłu maszynowego powołali zespół, który w terminie do 30 kwietnia ma przedłożyć raport oceniający rozwój polskiego przemysłu komputerowego. Powstaje wątpliwość, czy opracowanie takie będzie w pełni obiektywne. Liczne fakty wskazują bowiem na to, że właśnie wspomniane wyżej ministerstwa są odpowiedzialne za obecny, tragiczny stan przemysłu komputerowego — zarówno w sferze produkcji, jak i projektowania, badań oraz przygotowania ludzi. Można przypuszczać, że osoby odpowiedzialne za dotychczasowy stan rzeczy będą za wszelką cenę bronić osiągniętej pozycji osobistej. Nie można wykluczyć, że — chcąc siebie asekurować — posuną się do wypaczenia obrazu rzeczywistości, preparowania faktów oraz do rozpowszechniania frazesów o ogólnych trudnościach i cudownych perspektywach.

Podzielam istniejący pogląd, że zbyt wielu ludzi związanych z Ministerstwem Przemysłu Maszynowego i ze Zjednoczeniem MERA to mistrzowie pustosłowia, pozorów i niekompetencji. Takie fakty, jak podjęcie produkcji MERY 300 i zniszczenie K-202, bezsensowny eksport urządzeń peryferyjnych, niewytłumaczalne decyzje importowe oraz zmarnowanie wielkiego potencjału intelektualnego drzemiącego w istniejących placówkach naukowych — bardzo wymownie oskarżają!

Nikt nie może być sędzią w swojej sprawie, dlatego też należy jak najszybciej przedsięwziąć odpowiednie kroki, aby informatycy poznali fakty prawdziwe (i obiektywne ich naświetlenie) o stanie produkcji komputerów w kraju. Sprawdzone dane źródłowe powinny być zinterpretowane przez zespół n i e z a l e ż n y c h s p e c j a l i s t ó w. Ich opinie powinny pojawić się poza wszelką kolejnością na łamach INFORMATYKI. Sądzę, że to właśnie Redakcja powinna natychmiast zaproponować wybitnym znawcom przedmiotu, których autorytet moralny daje gwarancję obiektywności, aby w trybie pilnym przedstawili na łamach swój punkt widzenia. Wypowiedzi powinny dotyczyć przede wszystkim bieżącej sytuacji w dziedzinie produkcji i opracowań nowego sprzętu oraz najpilniejszych potrzeb krajowych.

Symbolem człowieka niosącego nadzieję (że nie wszystko jest stracone...) stał się p. inż. Jacek Karpiński. Nawet lokalna prasa codzienna szeroko omawia koleje jego życia. Wspomniano również, że ten wybitny konstruktor wysunął propozycję opracowania nowego komputera, który mógłby być przedmiotem eksportu; podobno toczone były rozmowy na temat warunków jego produkcji. Znając stan krajowej bazy podzespołów elektronicznych, mogę uznać za pewne, że produkcja maszyny cyfrowej o parametrach odpowiadających poziomowi światowemu musiałaby się oprzeć na elementach importowanych w większości z krajów zachodnich. Gdyby taką produkcję podjąć, to jest oczywiste, że w warunkach olbrzymich trudności gospodarczych sprzedaż w kraju nie wchodziłaby w rachubę.

Pytanie tylko, czy w obecnej sytuacji bezpośredni eksport komputerów jest najlepszym środkiem zdobywania dewiz. Sądzę, że należałoby wnikliwie rozważyć, czy podjęcie produkcji taniego i relatywnie dobrego sprzętu [1] — przy zminimalizowaniu importu części — nie byłaby bardziej uzasadniona ze społecznego punktu widzenia.

Można przypuszczać, że ożywienie krajowej informatyki dzięki dużym dostawom sprzętu mogłoby dać większy efekt ekonomiczny (i społeczny!), niż bezpośredni eksport sprzętu. Być może efekt ten byłby również wymierny dewizowo, np. przez poprawę jakości czy obniżkę kosztów wytwarzania eksportowanych dóbr. Można tu snuć pewne analogie z zastosowaniami mikroprocesorów do różnych

[1] nie pora na spory o nazewnictwo: nie jest ważne czy to będzie minikomputer, superkalkulator, mikrosystem — ważne, by dobrze funkcjonował

dokończenie na str. 40

Cena 30 zł